

[illegible][illegible]

CV
VO4[illegible]

(3)	77	EXTERNAL AND LOCAL DEFINITIONS
(4)	244	STANDARD TABLES
(7)	431	CONTROLLER INITIALIZATION ROUTINE
(8)	439	UNIT INITIALIZATION ROUTINE
(9)	503	DRIVER SPECIFIC SUBROUTINES
(10)	538	FDT ROUTINE - TEST TRANSFER BYTE COUNT ALIGNMENT
(11)	574	START I/O ROUTINE
(15)	1054	INTERRUPT SERVICE ROUTINE
(16)	1267	REGISTER DUMP ROUTINE


```
0000 1 .TITLE CVDRIVER - VAX/VMS VAX 8600 CONSOLE DISK DRIVER
0000 2 .IDENT 'V04-001'
0000 3
0000 4 .....
0000 5 .....
0000 6 *
0000 7 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 8 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 9 * ALL RIGHTS RESERVED.
0000 10 *
0000 11 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 12 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 13 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 14 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 15 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 16 * TRANSFERRED.
0000 17 *
0000 18 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 19 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 20 * CORPORATION.
0000 21 *
0000 22 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 23 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 24 *
0000 25 .....
0000 26 .....
0000 27 FACILITY:
0000 28
0000 29 VAX/VMS VAX 8600 CONSOLE RL02 DRIVER
0000 30
0000 31 AUTHOR:
0000 32
0000 33 BENN SCHREIBER, 15-MAR-1983
0000 34
0000 35 MODIFIED BY:
0000 36
0000 37 V04-001 BLS0345 Benn Schreiber 27-AUG-1984
0000 38 Retry complete transfer rather than attempting restart
0000 39 at last block. This avoids forking per-block in the
0000 40 non-error case. Wait for ready before issuing sts/reset
0000 41 on error path. Increase timeout on read/write operations.
0000 42 Check for errors on get status interrupts following read/write.
0000 43
0000 44 V03-005 BLS0342 Benn Schreiber 19-AUG-1984
0000 45 Implement abort, reset with status. Modify cvc_getsts
0000 46 to use TIMEDWAIT macro.
0000 47
0000 48 V03-004 TCM0002 Trudy C. Matthews 09-Aug-1984
0000 49 Increase timeout value in CVC_GETSTS from 30 to 100000.
0000 50
0000 51 V03-003 TCM0001 Trudy C. Matthews 08-Aug-1984
0000 52 Several bug fixes. Also a spec change - the LBN in STXCS
0000 53 must always be valid for each word of the transfer (and not
0000 54 just the first).
0000 55
0000 56 **
```

```
0000 58 : ABSTRACT:
0000 59 :
0000 60 : THIS MODULE CONTAINS THE TABLES AND ROUTINES NECESSARY TO
0000 61 : PERFORM ALL DEVICE-DEPENDENT PROCESSING OF AN I/O REQUEST
0000 62 : FOR RL02 DISK TYPES ON A VAX/VMS VAX 8600 CONSOLE SUBSYSTEM.
0000 63 :
0000 64 : THE DISKS HAVE THE FOLLOWING PHYSICAL GEOMETRY:
0000 65 :
0000 66 :           # CYL      TRACKS/   SECTORS/   BYTES/   MAXIMUM
0000 67 :           # CYL      CYLINDER  TRACK      SECTOR  BLOCKS
0000 68 :
0000 69 : RL02      512        2         40        256    20480
0000 70 :
0000 71 : THE IOSX INHSEEK FUNCTION MODIFIER IS TREATED AS A NO-OP BY
0000 72 : THIS DRIVER, SINCE AN IMPLICIT SEEK IS ALWAYS DONE BY THE
0000 73 : CONSOLE SUBSYSTEM WHEN READING/WRITING.
0000 74 :
0000 75 :--
```

```
0000 77 .SBTTL EXTERNAL AND LOCAL DEFINITIONS
0000 78
0000 79 :
0000 80 : EXTERNAL SYMBOLS
0000 81 :
0000 82 :
0000 83 $CRBDEF ;DEFINE CHANNEL REQUEST BLOCK
0000 84 $DCDEF ;DEFINE DEVICE CLASS
0000 85 $DDBDEF ;DEFINE DEVICE DATA BLOCK
0000 86 $DEVDEF ;DEFINE DEVICE CHARACTERISTICS
0000 87 $DPTDEF ;DEFINE DRIVER PROLOGUE TABLE
0000 88 $DYNDEF ;DEFINE DYNAMIC DATA STRUCTURE TYPES
0000 89 $EMBDEF ;DEFINE ERROR MESSAGE BUFFER
0000 90 $IDBDEF ;DEFINE INTERRUPT DATA BLOCK
0000 91 $IODEF ;DEFINE I/O FUNCTION CODES
0000 92 $IRPDEF ;DEFINE I/O REQUEST PACKET
0000 93 $PRDEF ;DEFINE PROCESSOR REGISTERS
0000 94 $SSDEF ;DEFINE SYSTEM STATUS CODES
0000 95 $UCBDEF ;DEFINE UNIT CONTROL BLOCK
0000 96 $VECDDEF ;DEFINE INTERRUPT VECTOR BLOCK
0000 97
0000 98 :
0000 99 : GENERATE CASE TABLE INDEX SYMBOL
0000 100 :
0000 101 .MACRO GENF FCODE
0000 102 CD'FCODE=$$GENF_CODE
0000 103 $$GENF_CODE=$$GENF_CODE+1
0000 104 .ENDM
0000 105 :
0000 106 : LOCAL SYMBOLS
0000 107 :
00000004 0000 108 CV_NUM_REGS = 4 ;NUMBER OF DEVICE REGISTERS (MIMIC RLO2)
00000005 0000 109 CV_SLM = 5 ;STATE=SEEK LINEAR MODE (READY TO GO)
0000 110 :
0000 111 : UCB OFFSETS WHICH FOLLOW THE STANDARD UCB FIELDS
0000 112 :
0000 113 $DEFINI UCB ;START OF UCB DEFINITIONS
0000 114 :
000000CC 0000 115 .=UCBSK_LCL_DISK_LENGTH ;BEGIN DEFINITIONS AT END OF UCB
00CC 116 $DEF UCB$S_LCV_CS .BLKL 1 ;CONTROL STATUS REGISTER
00D0 117 $DEF UCB$S_LCV_MP .BLKL 1 ;MULTIPURPOSE REGISTER
00D4 118 $DEF UCB$Q_LCV_CSMP .BLKQ 1 ;SAVE CS AND MP DURING RESET STATUS
00DC 119 $DEF UCB$B_LCV_STATE .BLKB 1 ;CURRENT INTERRUPT STATE **ADJACENCY
00DD 120 $DEF UCB$B_LCV_STS .BLKB 1 ;STATUS FLAGS **ASSUMED
00DE 121 $DEF UCB$W_LCV_BBC .BLKW 1 ;BLOCK BYTE COUNT REMAINING
00E0 122 $DEF UCB$S_LCV_IBUF .BLKL 1 ;INTERNAL BUFFER FOR READING
00E4 123 $DEF UCB$S_LCV_MVRTN .BLKL 1 ;ADDRESS OF BUFFER MOVE ROUTINE
00E8 124 $DEF UCB$S_LCV_BUFWIN .BLKL 1 ;BUFFER WINDOW
00EC 125 $DEF UCB$Q_LCV_BDAT .BLKQ 1 ;SAVPT AND TRANSFER PARAMS THIS BLOCK
00F4 126 $DEF UCB$S_LCV_LBN .BLKL 1 ;SAVE STARTING LBN OF TRANSFER
00F8 127 $DEF UCB$S_LCV_ABPC .BLKL 1 ;SAVE RETURN ADDRESS FROM ABORT CALL
00FC 128 $DEF UCB$K_LCV_LEN .BLKW 0 ;LENGTH OF UCB
00FC 129 $DEFEND UCB ;END OF UCB DEFINITIONS
0000 130 :
0000 131 :
0000 132 : RL11/RLO1 REGISTER OFFSETS FROM CSR ADDRESS
0000 133 :
```



```
0000 134 $DEFINI CV ; START OF REGISTER DEFINITIONS
0000 135
0000 136 :
0000 137 : UCSB_CV_STS FLAGS
0000 138 :
0000 139 _VIELD CV_0,<-
0000 140 <RD,,M>,- ;SET IF READ OPERATION
0000 141 <STSONLY,,M>,- ;OPERATION IS GET STATUS ONLY
0000 142 <STERROR,,M>,- ;ERROR FROM CONSOLE ON GETSTS INTERRUPT
0000 143 <ABORT,,M>> ;ABORT CURRENT OPERATION AND RETRY
0000 144
0000 145 $DEF CV_CS .BLKW 1 ;CONTROL STATUS REGISTER (CSR)
0002 146 _VIELD CV_CS_0,<- ;START OF CSR BIT DEFINITIONS
0002 147 <DRDY,,M>,- ; DRIVE READY
0002 148 <,3>,- ; FUNCTION CODE
0002 149 <,2>,- ; BUS ADDRESS EXTENSION BITS
0002 150 <,1>,- ; INTERRUPT ENABLE
0002 151 <,1>,- ; CONTROLLER READY
0002 152 <DS,,2>,- ; DRIVE SELECT
0002 153 <OPI,,M>,- ; OPERATION INCOMPLETE
0002 154 <CRC,,M>,- ; DATA CRC OR HEADER CRC
0002 155 <CVT,,M>,- ; DATA LATE OR HEADER NOT FOUND
0002 156 <NXM,,M>,- ; NON-EXISTENT MEMORY
0002 157 <DE,,M>,- ; DRIVE ERROR
0002 158 <CE,,M>,- ; COMPOSITE ERROR
0002 159 > ;END CSR BIT DEFINITIONS
0002 160
0002 161 $DEF CV_MP .BLKW 1 ;MULTIPURPOSE REGISTER (MPR)
0004 162 _VIELD CV_MP_0,<- ;START OF MPR BIT DEFINITIONS
0004 163 <STA,,3>,- ; DRIVE STATE
0004 164 <BH,,M>,- ; BRUSH HOME
0004 165 <HO,,M>,- ; HEADS OUT
0004 166 <CO,,M>,- ; COVER OPEN
0004 167 <HS,,M>,- ; HEAD SELECT
0004 168 <,1>,- ; DRIVE TYPE
0004 169 <DSE,,M>,- ; DRIVE SELECT ERROR
0004 170 <VC,,M>,- ; VOLUME CHECK
0004 171 <WGE,,M>,- ; WRITE GATE ERROR
0004 172 <SPE,,M>,- ; SPIN ERROR
0004 173 <SKTO,,M>,- ; SEEK TIME OUT
0004 174 <WL,,M>,- ; WRITE LOCK
0004 175 <CHE,,M>,- ; CURRENT HEAD ERROR
0004 176 <WDE,,M>,- ; WRITE DATA ERROR
0004 177 > ;END MPR BIT DEFINITIONS
0004 178
0004 179 :
0004 180 : VAX 8600 STXCS FORMAT
0004 181 :
0004 182 _VIELD STXCS_0,<-
0004 183 <FUNC,,4>,- ;DISK FUNCTION TO PERFORM
0004 184 <,2>,- ;MBZ
0004 185 <IE,,M>,- ;INTERRUPT ENABLE
0004 186 <RDY,,M>,- ;READY
0004 187 <ADDRS,,16>,- ;DISK LOGICAL BLOCK NUMBER
0004 188 <STS,,8>,- ;STATUS OF TRANSFER
0004 189 >
0004 190
```

```
0004 191          $DEFEND CV                      :END RL11/RL01 REGISTER DEFINITIONS
0000 192
0000 193 :
0000 194 : VAX 8600 CONSOLE STXCS STATUS CODES
0000 195 :
00000001 0000 196 TRANS_COMPLETE = 1                :TRANSACTION COMPLETED
00000002 0000 197 TRANS_CONTINUE = 2            :CONTINUE TRANSACTION
00000003 0000 198 TRANS_ABORTED = 3             :TRANSACTION ABORTED
00000004 0000 199 RETURNED_STATUS = 4           :STATUS RETURNED
00000080 0000 200 HANDSHAKE_ERROR = ^X80        :HANDSHAKE ERROR DURING TRANSACTION
00000081 0000 201 HW_ERROR = ^X81               :HARDWARE ERROR DURING TRANSACTION
0000 202 :
0000 203 : VAX 8600 CONSOLE STXCS FUNCTION CODES
0000 204 :
00000000 0000 205 NO_OP = 0                     :NO OPERATION
00000002 0000 206 STATUS_RESET = 2              :READ DEVICE STATUS WITH RST ASSERTED
00000003 0000 207 ABORT_TRANSFER = 3            :ABORT CURRENT TRANSFER
00000004 0000 208 READ_STATUS = 4              :READ DEVICE STATUS
00000005 0000 209 WRITE_BLOCK = 5              :WRITE BLOCK OF DATA
00000006 0000 210 READ_BLOCK = 6               :READ BLOCK OF DATA
0000 211 :
0000 212 : INTERRUPT TRANSITION CODES
0000 213 :
00000000 0000 214 ITC_DATA = 0                 :READ OR WRITE DATA
00000001 0000 215 ITC_STS1 = 1                 :GET CONTROL/STATUS REGISTER
00000002 0000 216 ITC_STS2 = 2                 :GET RL11 MULTIPURPOSE REGISTER
00000003 0000 217 ITC_ABORT = 3                :ABORT CURRENT TRANSFER
00000004 0000 218 ITC_RESET1 = 4               :GET CONTROL/STATUS WITH RST ASSERTED
00000005 0000 219 ITC_RESET2 = 5               :GET MP REG WITH RST ASSERTED
0000 220 :
0000 221 : DEFINE THE CASE OFFSETS AS CD'FUNCTION (I.E. CDF_NOP)
0000 222 :
00000000 0000 223 :                               $$GENF_CODE = 0                :INITIALIZE
0000 224 GENF F_NOP                               :NO-OP
0000 225 GENF F_UNLOAD                             :UNLOAD VOLUME (NOP)
0000 226 GENF F_SEEK                               :SEEK
0000 227 GENF F_RECAL                             :RECALIBRATE (NOP)
0000 228 GENF F_DRVCLR                             :DRIVE CLEAR (RESET & GET STATUS)
0000 229 GENF F_RELEASE                             :RELEASE PORT (NOP)
0000 230 GENF F_OFFSET                             :OFFSET HEADS (NOP)
0000 231 GENF F_RETCENTER                         :RETURN HEADS TO CENTERLINE (NOP)
0000 232 GENF F_PACKACK                           :PACK ACKNOWLEDGE (RESET & GET STATUS)
0000 233 GENF F_SEARCH                             :SEARCH (NOP)
0000 234 GENF F_WRITECHECK                         :WRITE CHECK
0000 235 GENF F_WRITEDATA                         :WRITE DATA
0000 236 GENF F_READDATA                          :READ DATA
0000 237 GENF F_WRITEHEAD                         :WRITE HEADERS (NOP)
0000 238 GENF F_READHEAD                         :READ HEADERS
0000 239 GENF F_NOP                               :place holder
0000 240 GENF F_NOP                               :place holder
0000 241 GENF F_AVAILABLE                         :AVAILABLE
0000 242
```



```
0000 244 .SBTTL STANDARD TABLES
0000 245
0000 246 :
0000 247 : DRIVER PROLOGUE TABLE
0000 248 :
0000 249 : THE DPT DESCRIBES DRIVER PARAMETERS AND I/O DATABASE FIELDS
0000 250 : THAT ARE TO BE INITIALIZED DURING DRIVER LOADING AND RELOADING
0000 251 :
0000 252 :
0000 253 DPTAB - ;DPT CREATION MACRO
0000 254 END=CV END,- ;END OF DRIVER LABEL
0000 255 ADAPTER=UBA,- ;ADAPTER TYPE = UNIBUS
0000 256 FLAGS=DPT$M_SVP,- ;SYSTEM PAGE TABLE ENTRY REQUIRED
0000 257 UCBSIZE=UCB$K_CV_LEN,- ;LENGTH OF UCB
0000 258 NAME=CVDRIVER ;DRIVER NAME
0038 259
0038 260 DPT_STORE INIT ;START CONTROL BLOCK INIT VALUES
0038 261 DPT_STORE DDB,DDB$S_ACPD,L,<^A\F11> ;DEFAULT ACP NAME
003F 262 DPT_STORE DDB,DDB$S_ACPD+3,B,DDB$K_CART ;ACP CLASS
0043 263 DPT_STORE UCB,UCB$B_FIPL,B,8 ;FORK IPL
0047 264 DPT_STORE UCB,UCB$S_DEVCHAR,L,- ;DEVICE CHARACTERISTICS
0047 265 <DEV$M_FOD- ;FILES ORIENTED
0047 266 :DEV$M_DIR- ;DIRECTORY STRUCTURED
0047 267 :DEV$M_AVL- ;AVAILABLE
0047 268 :DEV$M_ELG- ;ERROR LOGGING
0047 269 :DEV$M_SHR- ;SHAREABLE
0047 270 :DEV$M_IDV- ;INPUT DEVICE
0047 271 :DEV$M_ODV- ;OUTPUT DEVICE
0047 272 :DEV$M_RND> ;RANDOM ACCESS
004E 273 DPT_STORE UCB,UCB$B_DEVCLASS,B,DC$ DISK ;DEVICE CLASS
0052 274 DPT_STORE UCB,UCB$W_DEVBUFFS12,W,512 ;DEFAULT BUFFER SIZE
0057 275 DPT_STORE UCB,UCB$B_SECTORS,B,40 ;NUMBER OF SECTORS PER TRACK
005B 276 DPT_STORE UCB,UCB$B_TRACKS,B,2 ;NUMBER OF TRACKS PER CYLINDER
005F 277 DPT_STORE UCB,UCB$B_DIPL,B,21 ;DEVICE IPL
0063 278 DPT_STORE UCB,UCB$B_ERTMAX,B,8 ;MAX ERROR RETRY COUNT
0067 279
0067 280 DPT_STORE REINIT ;START CONTROL BLOCK RE-INIT VALUES
0067 281 DPT_STORE CRB,CRB$S_INTD+4,D,CV_INT ;INTERRUPT SERVICE ROUTINE ADDRESS
006C 282 DPT_STORE CRB,CRB$S_INTD+VEC$S_INITIAL,- ;CONTROLLER INIT ADDRESS
006C 283 D,CV_RL11_INIT
0071 284 DPT_STORE CRB,CRB$S_INTD+VEC$S_UNITINIT,- ;UNIT INIT ADDRESS
0071 285 D,CV_RLOX_INIT
0076 286 DPT_STORE DDB,DDB$S_DDT,D,CV$DDT ;DDT ADDRESS
007B 287
007B 288 DPT_STORE END ;END OF INITIALIZATION TABLE
0000 289
0000 290 :
0000 291 : DRIVER DISPATCH TABLE
0000 292 :
0000 293 : THE DDT LISTS ENTRY POINTS FOR DRIVER SUBROUTINES WHICH ARE
0000 294 : CALLED BY THE OPERATING SYSTEM.
0000 295 :
0000 296 :
0000 297 DDTAB - ;DDT CREATION MACRO
0000 298 DEVNAM=CV,- ;NAME OF DEVICE
0000 299 START=CV_STARTIO,- ;START I/O ROUTINE
0000 300 UNSOLIC=CV_UNSOINT,- ;UNSOLICITED INTERRUPT
```

```
0000 301      FUNCTB=CV_FUNC_TABLE,-      ;FUNCTION DECISION TABLE
0000 302      CANCEL=0,-                  ;CANCEL=NO-OP FOR FILES DEVICE
0000 303      REGDMP=CV_REGDUMP,-          ;REGISTER DUMP ROUTINE
0000 304      DIAGBF=<<<CV_NUM_REGS+5+5+3+1>*4>,- ;BYTES IN DIAG BUFFER
0000 305      ERLGBF=<<<<CV_NUM_REGS+5+1>*4>+EMBSL_DV_REGS+1> ;BYTES IN
0038 306                                     ;ERROR LOG BUFFER
0038 307
0038 308 : DIAGNOSTIC BUFFER SIZE = <<<4 RLO2 REGISTER LONGWORDS + 5 UCB FIELD LONGWORDS
0038 309      + 5 IOC$DIAGBUFILL LONGWORDS + 3 BUFFER ALLOCATION
0038 310      LONGWORDS + 1 LONGWORD FOR # REGISTERS IN CV_REGDUMP>
0038 311      * 4 BYTES/LONGWORD>
0038 312
0038 313 : ERROR LOG BUFFER SIZE = <<<<4 RLO2 REGISTER LONGWORDS + 5 UCB FIELD LONGWORDS
0038 314      + 1 LONGWORD FOR # REGISTERS IN CV_REGDUMP>
0038 315      * 4 BYTES/LONGWORD> + BYTES NEEDED FOR ERROR LOGGER
0038 316      TO SAVE SOFTWARE REGISTERS>
0038 317
0038 318
```

```
0038 320 :  
0038 321 : FUNCTION DECISION TABLE  
0038 322 :  
0038 323 : THE FDT LISTS VALID FUNCTION CODES, SPECIFIES WHICH  
0038 324 : CODES ARE BUFFERED, AND DESIGNATES SUBROUTINES TO  
0038 325 : PERFORM PREPROCESSING FOR PARTICULAR FUNCTIONS.  
0038 326 :  
0038 327 :  
0038 328 CV_FUNCTABLE:  
0038 329 FUNCTAB  
0038 330 :  
0038 331 <NOP,-  
0038 332 UNLOAD,-  
0038 333 SEEK,-  
0038 334 DRVCLR,-  
0038 335 PACKACK,-  
0038 336 SENSECHAR,-  
0038 337 SETCHAR,-  
0038 338 SENSEMODE,-  
0038 339 SETMODE,-  
0038 340 READLBLK,-  
0038 341 WRITELBLK,-  
0038 342 READPBLK,-  
0038 343 WRITEPBLK,-  
0038 344 READVBLK,-  
0038 345 WRITEVBLK,-  
0038 346 AVAILABLE,-  
0038 347 ACCESS,-  
0038 348 ACPCONTROL,-  
0038 349 CREATE,-  
0038 350 DEACCESS,-  
0038 351 DELETE,-  
0038 352 MODIFY,-  
0038 353 MOUNT-  
0038 354 >  
0040 354 FUNCTAB  
0040 355 :  
0040 356 <NOP,-  
0040 357 UNLOAD,-  
0040 358 SEEK,-  
0040 359 DRVCLR,-  
0040 360 PACKACK,-  
0040 361 SENSECHAR,-  
0040 362 SETCHAR,-  
0040 363 SENSEMODE,-  
0040 364 SETMODE,-  
0040 365 AVAILABLE,-  
0040 366 ACCESS,-  
0040 367 ACPCONTROL,-  
0040 368 CREATE,-  
0040 369 DEACCESS,-  
0040 370 DELETE,-  
0040 371 MODIFY,-  
0040 372 MOUNT-  
0040 373 >  
0048 373 FUNCTAB CV_ALIGN,-  
0048 374 <READLBLK,-  
0048 375 READPBLK,-  
0048 376 READVBLK,-
```

```
:LIST LEGAL FUNCTIONS  
: NO-OP  
: UNLOAD  
: SEEK  
: DRIVE CLEAR  
: PACK ACKNOWLEDGE  
: SENSE CHARACTERISTICS  
: SET CHARACTERISTICS  
: SENSE MODE  
: SET MODE  
: READ LOGICAL BLOCK  
: WRITE LOGICAL BLOCK  
: READ PHYSICAL BLOCK  
: WRITE PHYSICAL BLOCK  
: READ VIRTUAL BLOCK  
: WRITE VIRTUAL BLOCK  
: AVAILABLE  
: ACCESS FILE / FIND DIRECTORY ENTRY  
: ACP CONTROL FUNCTION  
: CREATE FILE AND/OR DIRECTORY ENTRY  
: DEACCESS FILE  
: DELETE FILE AND/OR DIRECTORY ENTRY  
: MODIFY FILE ATTRIBUTES  
: MOUNT VOLUME  
  
:BUFFERED FUNCTIONS  
: NO-OP  
: UNLOAD  
: SEEK  
: DRIVE CLEAR  
: PACK ACKNOWLEDGE  
: SENSE CHARACTERISTICS  
: SET CHARACTERISTICS  
: SENSE MODE  
: SET MODE  
: AVAILABLE  
: ACCESS FILE / FIND DIRECTORY ENTRY  
: ACP CONTROL FUNCTION  
: CREATE FILE AND/OR DIRECTORY ENTRY  
: DEACCESS FILE  
: DELETE FILE AND/OR DIRECTORY ENTRY  
: MODIFY FILE ATTRIBUTES  
: MOUNT VOLUME  
  
:TEST ALIGNMENT FUNCTIONS  
: READ LOGICAL BLOCK  
: READ PHYSICAL BLOCK  
: READ VIRTUAL BLOCK
```


0048	377	WRITEBLK,-	: WRITE LOGICAL BLOCK
0048	378	WRITEPBLK,-	: WRITE PHYSICAL BLOCK
0048	379	WRITEVBLK-	: WRITE VIRTUAL BLOCK
0048	380	>	
0054	381	FUNCTAB +ACPSREADBLK,-	: READ FUNCTIONS
0054	382	<READLBLK,-	: READ LOGICAL BLOCK
0054	383	READPBLK,-	: READ PHYSICAL BLOCK
0054	384	READVBLK-	: READ VIRTUAL BLOCK
0054	385	>	
0060	386	FUNCTAB +ACPSWRITEBLK,-	: WRITE FUNCTIONS
0060	387	<WRITEBLK,-	: WRITE LOGICAL BLOCK
0060	388	WRITEPBLK,-	: WRITE PHYSICAL BLOCK
0060	389	WRITEVBLK-	: WRITE VIRTUAL BLOCK
0060	390	>	
006C	391	FUNCTAB +ACPSACCESS,-	: ACCESS FUNCTIONS
006C	392	<ACCESS,-	: ACCESS FILE / FIND DIRECTORY ENTRY
006C	393	CREATE-	: CREATE FILE AND/OR DIRECTORY ENTRY
006C	394	>	
0078	395	FUNCTAB +ACPSDEACCESS,-	: DEACCESS FUNCTION
0078	396	<DEACCESS-	: DEACCESS FILE
0078	397	>	
0084	398	FUNCTAB +ACPSMODIFY,-	: MODIFY FUNCTIONS
0084	399	<ACPCONTROL,-	: ACP CONTROL FUNCTION
0084	400	DELETE,-	: DELETE FILE AND/OR DIRECTORY ENTRY
0084	401	MODIFY-	: MODIFY FILE ATTRIBUTES
0084	402	>	
0090	403	FUNCTAB +ACPSMOUNT,-	: MOUNT FUNCTION
0090	404	<MOUNT-	: MOUNT VOLUME
0090	405	>	
009C	406	FUNCTAB +EXESLCLDSKVALID,-	
009C	407	<UNLOAD,-	
009C	408	AVAILABLE,-	
009C	409	PACKACK-	
009C	410	>	
00A8	411	FUNCTAB +EXESZEROPARM,-	: ZERO PARAMETER FUNCTIONS
00A8	412	<NOP,-	: NO-OP
00A8	413	UNLOAD,-	: UNLOAD
00A8	414	DRVCLR,-	: DRIVE CLEAR
00A8	415	PACKACK,-	: PACK ACKNOWLEDGE
00A8	416	AVAILABLE,-	: AVAILABLE
00A8	417	>	
00B4	418	FUNCTAB +EXESONEPARM,-	: ONE PARAMETER FUNCTION
00B4	419	<SEEK-	: SEEK
00B4	420	>	
00C0	421	FUNCTAB +EXESSENSEMODE,-	: SENSE FUNCTIONS
00C0	422	<SENSECHAR,-	: SENSE CHARACTERISTICS
00C0	423	SENSEMODE-	: SENSE MODE
00C0	424	>	
00CC	425	FUNCTAB +EXESSETCHAR,-	: SET FUNCTIONS
00CC	426	<SETCHAR,-	: SET CHARACTERISTICS
00CC	427	SETMODE-	: SET MODE
00CC	428	>	

```
00D8 430
00D8 431      .SBTTL  CONTROLLER INITIALIZATION ROUTINE
00D8 432      ++
00D8 433
00D8 434      FUNCTIONAL DESCRIPTION:
00D8 435
00D8 436      THIS ROUTINE IS A NO-OP FOR THE RL11 BUT MUST BE INCLUDED
00D8 437      SINCE IT IS CALLED WHEN THE RL02 IS BOOTED AS A SYSTEM DEVICE.
00D8 438
00D8 439      THE OPERATING SYSTEM CALLS THIS ROUTINE:
00D8 440      - AT SYSTEM STARTUP
00D8 441      - DURING DRIVER LOADING
00D8 442      - DURING RECOVERY FROM POWER FAILURE
00D8 443
00D8 444      INPUTS:
00D8 445
00D8 446      R4      - CSR ADDRESS (DEVICE CONTROL STATUS REGISTER)
00D8 447      R5      - IDB ADDRESS (INTERRUPT DATA BLOCK)
00D8 448      ALL INTERRUPTS ARE LOCKED OUT
00D8 449
00D8 450      OUTPUTS:
00D8 451
00D8 452      CONTROL IS RETURNED TO THE CALLER.
00D8 453
00D8 454      --
00D8 455
05 00D8 456 CV_RL11_INIT:      ;CONTROLLER INITIALIZATION
00D8 457      RSB              ;RETURN TO CALLER
```

```
00D9 459 .SBTTL UNIT INITIALIZATION ROUTINE
00D9 460
00D9 461 :++
00D9 462
00D9 463 CV_RLOX_INIT - UNIT INITIALIZATION ROUTINE
00D9 464
00D9 465 FUNCTIONAL DESCRIPTION:
00D9 466
00D9 467 THIS ROUTINE READIES THE RLO2 UNIT FOR I/O OPERATIONS.
00D9 468
00D9 469 THE OPERATING SYSTEM CALLS THIS ROUTINE:
00D9 470 - AT SYSTEM STARTUP
00D9 471 - DURING DRIVER LOADING
00D9 472 - DURING RECOVERY FROM POWER FAILURE
00D9 473
00D9 474 INPUTS:
00D9 475
00D9 476 R4 - CSR ADDRESS (CONTROLLER STATUS REGISTER)
00D9 477 R5 - UCB ADDRESS (UNIT CONTROL BLOCK)
00D9 478
00D9 479 OUTPUTS:
00D9 480
00D9 481 THE DRIVE UNIT IS RESET, UCB FIELDS ARE INITIALIZED, AND THE
00D9 482 ROUTINE WAITS FOR ONLINE UNITS TO SPIN UP. ALL REGISTERS
00D9 483 EXCEPT R0-R3 ARE PRESERVED.
00D9 484
00D9 485 :--
00D9 486
00D9 487 CV_RLOX_INIT:
00D9 488 BICW2 #<UCBSM_ONLINE!UCBSM_VALID>,- ;RLO1/RLO2 UNIT INITIALIZATION
00DD 489 UCBSM_STS(R5) ;ASSUME OFFLINE/INVALID
00DF 490
00DF 491 MOVL #X2324C002,- ;SET MEDIA IDENT 'DL RLO2'
008C C5 008C C5 00E5 492 UCBSL_MEDIA_ID(R5)
00E8 493 MOV B S^#DTS_RLO2,- ;SET RLO2 DEVICE TYPE
00EA 494 UCBSB_DEVTYPE(R5)
00EC 495 MOVW #512,UCBSW_CYLINDERS(R5);SET NUMBER OF RLO2 CYLINDERS
00B0 C5 5000 8F 3C 00F2 496 MOVZWL #20480,UCBSL_MAXBLOCK(R5);SET MAX RLO2 BLOCK NUMBER
00F9 497 BSBB CVC_GETSTS ;GET CONSOLE RLO2 STATUS
00FB 498 BBC #CV-CS_VDRDY,R0,40$ ;BRANCH IF DRIVE NOT READY
00FF 499 BISW2 #UCBSM_VALID,UCBSW_STS(R5);YES, SET VOLUME VALID
0105 500 40$: BISW2 #UCBSM_ONLINE,UCBSW_STS(R5);SET UNIT ONLINE
0109 501 60$: RSB
```

0810 8F AA 00D9 488
64 A5 00DD 489
2324C002 8F D0 00DF 491
008C C5 00E5 492
41 A5 00E8 493
46 A5 0200 8F B0 00EA 494
00B0 C5 5000 8F 3C 00F2 496
06 50 00 E1 00F9 497
64 A5 0800 8F A8 00FB 498
64 A5 10 A8 00FF 499
05 0105 500 40\$:
0109 501 60\$: RSB


```
010A 503 .SBTTL DRIVER SPECIFIC SUBROUTINES
010A 504
010A 505 CVC_GETSTS - GET STATUS FOR VAX 8600 CONSOLE RLO2 WITHOUT INTERRUPTS
010A 506
010A 507 INPUTS:
010A 508
010A 509 NONE
010A 510
010A 511 OUTPUTS:
010A 512
010A 513 R0 = 0 IF FAILED TO GET STATUS
010A 514 = RLO2 CONTROL STATUS REGISTER
010A 515
010A 516 R1 = RLO2 MULTIPURPOSE REGISTER (UNUSABLE IF R0=0)
010A 517
010A 518 CVC_GETSTS:
52 DD 010A 519 PUSHL R2 ;SAVE R2
0D 10 010C 520 BSBB 100$ ;READ CONTROL STATUS REGISTER
50 DD 010E 521 PUSHL R0 ;SAVE R0
09 10 0110 522 BSBB 100$ ;READ MULTIPURPOSE REGISTER
S1 50 D0 0112 523 MOVL R0,R1 ;POSITION MULTIPURPOSE REGISTER
50 8ED0 0115 524 POPL R0 ;RESTORE CSR
04 BA 0118 525 POPR #^M<R2> ;RESTORE R2
05 011A 526 RSB
0000004C 8F 02 DA 011B 527
0122 528 100$: MTPR #STATUS RESET,#PRS_STXCS ;REQUEST READ STATUS
0122 529 TIMEDWAIT TIME=#600*1000,-
0122 530 INS1=<MFPR #PRS_STXCS,R2>,- ;READ STATUS
05 014F 531 INS2=<BBS #STXCS_V_RDY,R2,140$> ;BRANCH IF READY
0150 532 RSB ;CONSOLE NEVER GOT READY (TIMEDWAIT
533 ;CLEAR R0 ON FALL-OUT)
SE 04 C0 0150 534 140$: ADDL2 #4,SP ;CLEAR TIMEDWAIT'S COUNTER FROM STACK
50 0000004D 8F DB 0153 535 MFPR #PRS_STXDB,R0 ;OBTAIN STATUS FROM CONSOLE
05 015A 536 RSB
```

```
0158 538 .SBTTL FDT ROUTINE - TEST TRANSFER BYTE COUNT ALIGNMENT
0158 539
0158 540 :++
0158 541
0158 542 DL_ALIGN - FDT ROUTINE TO TEST XFER BYTE COUNT
0158 543
0158 544 FUNCTIONAL DESCRIPTION:
0158 545
0158 546 THIS ROUTINE IS CALLED FROM THE FUNCTION DECISION TABLE DISPATCHER
0158 547 TO CHECK THE BYTE COUNT PARAMETER SPECIFIED BY THE USER PROCESS
0158 548 FOR AN EVEN NUMBER OF BYTES (WORD BOUNDARY).
0158 549
0158 550 INPUTS:
0158 551
0158 552 R3 - IRP ADDRESS (I/O REQUEST PACKET)
0158 553 R4 - PCB ADDRESS (PROCESS CONTROL BLOCK)
0158 554 R5 - UCB ADDRESS (UNIT CONTROL BLOCK)
0158 555 R6 - CCB ADDRESS (CHANNEL CONTROL BLOCK)
0158 556 R7 - BIT NUMBER OF THE I/O FUNCTION CODE
0158 557 R8 - ADDRESS OF FDT TABLE ENTRY FOR THIS ROUTINE
0158 558 4(AP) - ADDRESS OF FIRST FUNCTION DEPENDENT QIO PARAMETER
0158 559
0158 560 OUTPUTS:
0158 561
0158 562 IF THE QIO BYTE COUNT PARAMETER IS ODD, THE I/O OPERATION IS
0158 563 TERMINATED WITH AN ERROR. IF IT IS EVEN, CONTROL IS RETURNED
0158 564 TO THE FDT DISPATCHER.
0158 565
0158 566 :--
0158 567
0158 568 CV_ALIGN:
0158 569 BLBS 4(AP),10$ ;CHECK BYTE COUNT AT P1(AP)
0158 570 RSB ;IF LBS - ODD BYTE COUNT
0158 571 10$: MOVZWL #SS$ IVBUFLN,R0 ;EVEN - RETURN TO CALLER
0158 572 JMP G*EX$ABORTIO ;SET BUFFER ALIGNMENT STATUS
;ABORT I/O
```

01 04 AC E8
50 034C 8F 3C
00000000'GF 17

```
0168 574 .SBTTL START I/O ROUTINE
0168 575
0168 576 :++
0168 577
0168 578 CV_STARTIO - START I/O ROUTINE
0168 579
0168 580 FUNCTIONAL DESCRIPTION:
0168 581
0168 582 THIS FORK PROCESS IS ENTERED FROM THE EXECUTIVE AFTER AN I/O REQUEST
0168 583 PACKET HAS BEEN DEQUEUED, AND PERFORMS THE FOLLOWING:
0168 584
0168 585 - ACTIVATES THE CONSOLE AFTER SETTING UCB FIELDS, AND OBTAINING
0168 586 CONTROLLER RESOURCES
0168 587
0168 588 - WAITS FOR AN INTERRUPT
0168 589
0168 590 - REGAINS CONTROL AFTER THE ISR SERVICES THE INTERRUPT, AND
0168 591 - RE-ACTIVATES THE CONSOLE IF THE ORIGINAL FUNCTION
0168 592 IS A RETRIABLE ERROR, OR
0168 593 - COMPLETES THE I/O REQUEST BY RELEASING RESOURCES,
0168 594 SETTING STATUS CODES, AND RETURNING TO THE EXECUTIVE.
0168 595
0168 596 INPUTS:
0168 597
0168 598 R3 - IRP ADDRESS (I/O REQUEST PACKET)
0168 599 R5 - UCB ADDRESS (UNIT CONTROL BLOCK)
0168 600
0168 601 OUTPUTS:
0168 602
0168 603 R0 - FIRST I/O STATUS LONGWORD: STATUS CODE & BYTES XFERED
0168 604 R1 - SECOND I/O STATUS LONGWORD: 0 FOR DISKS
0168 605
0168 606 THE I/O FUNCTION IS EXECUTED.
0168 607
0168 608 ALL REGISTERS EXCEPT R0-R4 ARE PRESERVED.
0168 609
0168 610 :--
0168 611
0168 612 CV_STARTIO: ;START I/O OPERATION
0168 613
0168 614 PREPROCESS UCB FIELDS
0168 615
0168 616 PREPROCESS:
0168 617
0168 618 Convert the physical media address in IRPSL_MEDIA to an LBN.
0168 619 This is necessary because the console RL02 controller expects an LBN,
0168 620 not a physical media address. The LBN is given by the formula:
0168 621
0168 622 LBN = (CYLINDER*(TRACKS/CYLINDER)+TRACK)*(SECTORS/TRACK)+SECTOR
0168 623
0168 624 MOVL IRPSL_MEDIA(R3),- ;Copy media address to UCB
0168 625 UCBSL_MEDIA(R5)
0168 626 MOVAL UCBSL_MEDIA(R5),R3 ;Get address of media address
0168 627 MOVZBL (R3)+,R0 ;Get SECTOR
0168 628 MOVZBL (R3)+,R1 ;Get TRACK
0168 629 MOVZWL (R3)+,-(SP) ;Get CYLINDER
0168 630 MOVZBL UCBSL_TRACKS(R5),R3 ;Get TRACKS/CYLINDER
```

38	A3	D0	0168	624
00BC	C5		016E	625
53	00BC	C5	DE	0171
50	83	9A	0176	627
51	83	9A	0179	628
7E	83	3C	017C	629
53	45	A5	9A	017F


```

      53 51 44 A5 9A 0189 633
      51 51 51 C4 018D 634
00BC C5 50 51 C1 0190 635
00F4 C5 00BC C5 D0 0196 636
      00B1 C5 90 019D 637
      00B0 C5 01A1 638
      53 58 A5 D0 01A4 639
009A C5 20 A5 B0 01A8 640
      00 00 EF 01AE 641
      51 20 A5 06 01B0 642
      0092 C5 51 90 01B4 643
      78 A5 7D 01B9 644
      00EC C5 01BC 645
      02 AA 01BF 646
      68 A5 01C1 647
      07 E1 01C3 648
      04 2A A5 01C5 649
      68 A5 02 AB 01C8 650
      01CC 651
      01CC 652
      01CC 653
      01CC 654
      01CC 655
      01CC 656
      01CC 657
      01CC 658
00C0 C5 7E A5 AE 01CC 659
      00EC C5 7D 01D2 660
      78 A5 01D6 661
00BC C5 00F4 C5 D0 01D8 662
      01DF 663
      53 58 A5 D0 01DF 664
      08 E0 01E3 665
      0D 2A A5 01E5 666
      08 E0 01E8 667
      08 64 A5 01EA 668
      50 0254 BF 3C 01ED 669
      0240 31 01F2 670
      53 0092 C5 9A 01F5 671
      00DC C5 B4 01FA 672
      00EB C5 D4 01FE 673
      0202 674
      0202 675
      0202 676
      0202 677
      0202 678
      0202 679
      0202 680
      0202 681
      0202 682
      0202 683
      0202 684
      0202 685
      0202 686
      0202 687

MULL2 (SP)+,R3 ;R3 = (T/C)
ADDL2 R3,R1 ;R1 = (T/C)+T
MOVZBL UCBSB_SECTORS(R5),R3 ;Get SECTORS/TRACK
MULL2 R3,R1 ;R1 = ((T/C)+T)*(S/T)
ADDL3 R1,R0,UCBSL_MEDIA(R5) ;CALULATE AND STORE LBN
MOVL UCBSL_MEDIA(R5),UCBSL_CV_LBN(R5) ;SAVE STARTING LBN FOR RETRIES
MOVW UCBSB_ERTMAX(R5),- ;INITIALIZE ERROR RETRY COUNT
      UCBSB_ERTCNT(R5)
      ;
MOVL UCBSL_IRP(R5),R3 ;GET IRP ADDRESS
MOVW IRPSW_FUNC(R3),UCBSW_FUNC(R5) ;SAVE FUNCTION CODE
EXTZV #IRPSW_FCODE,- ;EXTRACT I/O FUNCTION CODE
      #IRPSW_FCODE,IRPSW_FUNC(R3),R1
MOVW R1,UCBSB_FEX(R5) ;STORE FUNCTION DISPATCH INDEX
MOVW UCBSL_SVAPTE(R5),- ;SAVE TRANSFER PARAMETERS
      UCBSQ_CV_BDAT(R5)
BICW2 #UCBSM_DIAGBUF,- ;CLR DIAGNOSTIC BUFFER PRESENT
      UCBSW_DEVSTS(R5) ;IF CLR - NO DIAG BUFFER
BBC #IRPSW_DIAGBUF,-
      IRPSW_STS(R3),FDISPATCH
BISW2 #UCBSM_DIAGBUF,UCBSW_DEVSTS(R5) ;SET DIAG BUFFER PRESENT

      ;
      CENTRAL FUNCTION DISPATCH
      ;
      FDISPATCH: ;FUNCTION DISPATCH
      ;
      ; RETRY LOGIC IS DONE BY RESTARTING THE ENTIRE TRANSFER, RATHER THAN
      ; AT THE BLOCK IN ERROR. HENCE, WE RESTORE TRANSFER PARAMETERS HERE
      ;
      MNEGW UCBSW_BCNT(R5),UCBSW_BCR(R5) ;INIT NEG BYTES LEFT TO XFER
      MOVW UCBSQ_CV_BDAT(R5),- ;RESTORE TRANSFER PARAMETERS
      UCBSL_SVAPTE(R5)
      MOVL UCBSL_CV_LBN(R5),UCBSL_MEDIA(R5) ;RESTORE STARTING LBN
      ;
      MOVL UCBSL_IRP(R5),R3 ;GET IRP ADDRESS
      BBS #IRPSW_PHYSIO,- ;IF SET - PHYSICAL I/O FUNCTION
      IRPSW_STS(R3),108
      BBS #UCBSW_VALID,- ;IF SET - VOLUME SOFTWARE VALID
      UCBSW_STS(R5),108
      MOVZWL #SSS_VOLINV,R0 ;SET VOLUME INVALID STATUS
      BRW RESETXFR ;RESET BYTE COUNT AND EXIT
      MOVZBL UCBSB_FEX(R5),R3 ;GET FUNCTION DISPATCH INDEX
      CLRW UCBSB_CV_STAT(R5) ;CLEAR INTERRUPT STATE AND STATUS
      CLRL UCBSL_CV_BUFWIN(R5) ;CLEAR BUFFER WINDOW
      CASE R3,- ;DISPATCH TO FUNCTION HANDLING ROUTINE
      NOP ;NOP
      UNLOAD,- ;UNLOAD
      SEEK,- ;SEEK
      NOP,- ;RECALIBRATE (unsupported)
      DRVCLR,- ;DRVCLR
      NOP,- ;RELEASE PORT (unsupported)
      NOP,- ;OFFSET HEADS (unsupported)
      NOP,- ;RETURN TO CENTER (unsupported)
      PACKACK,- ;PACK ACKNOWLEDGE
      NOP,- ;SEARCH (unsupported)
      WRITECHECK,- ;WRITE CHECK (unsupported)
      WRITEDATA,- ;WRITE DATA
      READDATA,- ;READ DATA
      108:

```

```
0202 688 NOP,- ; WRITE HEADER (unsupported)
0202 689 NOP,- ; READ HEADER
0202 690 NOP,- ; place holder
0202 691 NOP,- ; place holder
0202 692 AVAILABLE- ; AVAILABLE
0202 693 >
022A 694
022A 695 NOP: ;NO-OP
022A 696 WRITECHECK: ;WRITE CHECK
022A 697 SEEK: ;SEEK
022A 698 DRVCLR: ;DRIVE CLEAR (GET STATUS & RESET)
022A 699 DO_FUNCTION:
022A 700 CLR8 UCBSB_FEX(R5) ;SET FUNCTION
0092 C5 94 022E 701 BSBW FEXL ;EXECUTE FUNCTION
0097 30 0231 702 .BYTE RETRYERR--1 ;ERROR OFFSET
19 11 0232 703 BRB NORMAL ;DONE
0234 704
0234 705 PACKACK: ;PACK ACKNOWLEDGE (GET STATUS & RESET)
64 A5 0800 8F AB 0234 706 BISM2 #UCBSM_VALID, - ;Set software volume valid bit.
EE 11 023A 707 BRB UCBSW_STS(R5)
023C 708 DO_FUNCTION ;Then go do hardware function.
023C 710 UNLOAD: ;UNLOAD
023C 711 AVAILABLE: ;AVAILABLE
64 A5 0800 8F AA 023C 712 BICW2 #UCBSM_VALID, - ;Clear software volume valid bit.
09 11 0242 713 BRB UCBSW_STS(R5) ;and go complete operation without
0244 714 NORMAL ;any hardware interaction.
0244 715
0244 716 READDATA: ;READ DATA
00DD C5 01 88 0244 717 BISB #CV_M_RD,UCBSB_CV_STS(R5) ;SET READ FLAG
0249 718 WRITEDATA: ;WRITE DATA
007C 30 0249 719 BSBW FEXL ;EXECUTE FUNCTION
05 024C 720 .BYTE RETRYERR--1 ;ERROR OFFSET
024D 721 :
024D 722 : OPERATON COMPLETION
024D 723 :
024D 724
024D 725 NORMAL: ;SUCCESSFUL OPERATION COMPLETE
50 01 3C 024D 726 MOVZWL #SSB_NORMAL,R0 ;SET NORMAL COMPLETION STATUS
43 11 0250 727 BRB FUNCXT ;FUNCTION EXIT
0252 728
0252 729 RETRYERR: ;RETRIABLE ERROR
0080 C5 97 0252 730 DECB UCBSB_ERTCNT(R5) ;ANY RETRIES LEFT?
03 13 0256 731 BEQL FATALERR ;IF EQL - NO
FF71 31 0258 732 BRW FDISPATCH ;RETRY FUNCTION
025B 733
025B 734 FATALERR: ;UNRECOVERABLE ERROR
50 0254 8F 3C 025B 735 MOVZWL #SSB_VOLINV,R0 ;ASSUME VOLUME INVALID STATUS
51 00CC C5 7D 0260 736 MOVQ UCBSB_CV_CS(R5),R1 ;GET CS IN R1 AND MP IN R2
2C 52 09 E0 0265 737 BBS #CV_MP_V_VC,R2,FUNCXT ;IF SET - VOLUME INVALID
0269 738
50 025C 8F 3C 0269 739 MOVZWL #SSB_WRTLCK,R0 ;ASSUME WRITE LOCK ERROR STATUS
04 52 0D E1 026E 740 BBC #CV_MP_V_WL,R2,10B ;IF CLR - VOLUME NOT WRITE LOCKED
1F 52 0A E0 0272 741 BBS #CV_MP_V_WGE,R2,FUNCXT ;IF SET - WRITE GATE ERROR
0276 742 ;IF WL & WGE SET - WRITE LOCK ERROR
0276 743
50 01F4 8F 3C 0276 744 10B: MOVZWL #SSB_PARITY,R0 ;ASSUME PARITY ERROR STATUS
```

16	S1	0B	E0	027B	745	BBS	#CV_CS_V_CRC,R1,FUNCXT	:IF SET - CRC ERROR
				027F	746			:OR DATAPATH PURGE ERROR
04	S1	0A	E1	027F	747	BBC	#CV_CS_V_OPI,R1,208	:HEADER NOT FOUND ERROR?
0E	S1	0C	E0	0283	748	BBS	#CV_CS_V_CVT,R1,FUNCXT	:IF OPI AND CVT SET - YES
				0287	749			
50	008C	BF	3C	0287	750	MOVZWL	#SSB_DRVERR,R0	:ASSUME DRIVE ERROR STATUS
05	S1	0E	E0	028C	751	BBS	#CV_CS_V_DE,R1,FUNCXT	:IF SET - DRIVE ERROR
				0290	752			
50	0054	BF	3C	0290	753	MOVZWL	#SSB_CTRLERR,R0	:ASSUME CONTROLLER ERROR STATUS
				0295	754			
				0295	755	FUNCXT:		:FUNCTION EXIT
				0295	756	PUSHL	R0	:SAVE FINAL REQUEST STATUS
U0000000	GF	16	D0	0297	757	JSB	G*IOC\$DIAGBUFILL	:FILL DIAGNOSTIC BUFFER IF PRESENT
0092	C5	0A	91	029D	758	CMPB	#CDF_WRITECHECK,UCBSB_FEX(R5)	:DRIVE RELATED FUNCTION?
		13	1A	02A2	759	BGTRU	108	:IF GTRU - YES
0092	C5	11	91	02A4	760	CMPB	#CDF_AVAILABLE,UCBSB_FEX(R5)	:DRIVE RELATED FUNCTION?
		0C	13	02A9	761	BEQL	108	:IF EQL - YES
53	58	A5	D0	02AB	762	MOVL	UCBSL_IRP(R5),R3	:RETRIEVE ADDRESS OF IRP
	00C0	C5	A1	02AF	763	ADDW3	UCBSW_BCR(R5),-	:CALCULATE BYTES TRANSFERRED
02	AE	32	A3	02B3	764		IRPSW_BCNT(R3),2(SP)	:
				02B7	765	108:	RELCHAN	:RELEASE CHANNEL IF OWNED
				02BD	766			
				02BD	767	CLRL	R1	:CLEAR SECOND STATUS LONGWORD
				02BF	768	POPL	R0	:RETRIEVE FINAL REQUEST STATUS
				02C2	769	REQCOM		:COMPLETE REQUEST


```
02C8 771  
02C8 772  
02C8 773  
02C8 774  
02C8 775  
02C8 776  
02C8 777  
02C8 778  
02C8 779  
02C8 780  
02C8 781  
02C8 782  
02C8 783  
02C8 784  
02C8 785  
02C8 786  
02C8 787  
02C8 788  
02C8 789  
02C8 790  
02C8 791  
02C8 792  
02C8 793  
02C8 794  
02C8 795  
02C8 796  
02C8 797  
02C8 798  
02C8 799  
02C8 800  
02C8 801  
02C8 802  
02C8 803  
02C8 804  
02C8 805  
02C8 806  
02C8 807  
02C8 808  
02C8 809  
02C8 810  
02C8 811  
02C8 812  
02C8 813  
02C8 814  
02C8 815  
02C8 816  
02C8 817  
02C8 818  
02C8 819  
02C8 820  
02C8 821  
02C8 822  
02C8 823  
02C8 824  
02C8 825  
02C8 826  
02D1 827
```

FEXL - RL11 HARDWARE FUNCTION EXECUTION

THIS ROUTINE IS CALLED VIA A BSB WITH A BYTE IMMEDIATELY FOLLOWING THAT SPECIFIES THE ADDRESS OF AN ERROR ROUTINE. ALL DATA IS ASSUMED TO HAVE BEEN SET UP IN THE UCB BEFORE THE CALL. THE APPROPRIATE PARAMETERS ARE LOADED INTO THE CONSOLE STXCS AND THE FUNCTION IS INITIATED. THE RETURN ADDRESS IS STORED IN THE UCB AND A WAITFOR INTERRUPT IS EXECUTED. WHEN THE INTERRUPT OCCURS, CONTROL IS RETURNED TO THE CALLER.

INPUTS:

R5 = DEVICE UNIT UCB ADDRESS

00(SP) = RETURN ADDRESS OF CALLER
04(SP) = RETURN ADDRESS OF CALLER'S CALLER

IMMEDIATELY FOLLOWING INLINE AT THE CALL SITE IS A BYTE WHICH CONTAINS A BRANCH DESTINATION TO AN ERROR RETRY ROUTINE.

OUTPUTS:

THERE ARE FOUR EXITS FROM THIS ROUTINE:

1. SPECIAL CONDITION - THIS EXIT IS TAKEN IF A POWER FAILURE OCCURS OR THE OPERATION TIMES OUT. IT IS A JUMP TO THE APPROPRIATE ERROR ROUTINE. IN THE CASE OF A POWER FAILURE, THE RETRY COUNT IS RESET AND RETRIES INITIATED. IN THE CASE OF A TIMEOUT, THE RETRY COUNT IS DECREMENTED AND RETRIES INITIATED IF RETRIES REMAIN.
2. FATAL ERROR - THIS EXIT IS TAKEN IF A FATAL CONTROLLER OR DRIVE ERROR OCCURS OR IF ANY ERROR OCCURS AND ERROR RETRY IS EITHER INHIBITED OR EXHAUSTED. IT IS A JUMP TO THE FATAL ERROR EXIT ROUTINE.
3. RETRIABLE ERROR - THIS EXIT IS TAKEN IF A RETRIABLE CONTROLLER OR DRIVE ERROR OCCURS AND ERROR RETRY IS NEITHER INHIBITED NOR EXHAUSTED. IT CONSISTS OF TAKING THE ERROR BRANCH EXIT SPECIFIED AT THE CALL SITE. RETRIES ARE ACCOMPLISHED BY RESTARTING THE ENTIRE I/O OPERATION, RATHER THAN AT THE BLOCK IN ERROR.
4. SUCCESSFUL OPERATION - THIS EXIT IS TAKEN IF NO ERRORS OCCUR DURING THE OPERATION. IT CONSISTS OF A RETURN INLINE.

IN ALL CASES IF AN ERROR OCCURS, AN ATTEMPT IS MADE TO LOG THE ERROR.

IN ALL CASES FINAL DEVICE REGISTERS ARE RETURNED VIA THE UCB.

UCBSW_BCR(R5) = NEGATIVE BYTES REMAINING TO TRANSFER

FEXL:

```
009C C5 BED0  
50 24 A5 DO  
51 2C A0 DO
```

```
POPL UCB$DPC(R5) ;FUNCTION EXECUTOR  
MOVL UCB$LCRB(R5),R0 ;SAVE DRIVER PC VALUE  
MOVL CRB$INTD+VE($L_IDB(R0),R1 ;GET ADDRESS OF PRIMARY CRB  
;GET ADDRESS OF IDB
```

```
04 A1 55 D1 02D5 828 CMPL R5, IDB$$_OWNER(R1) ; DOES THIS PROCESS OWN CHANNEL?
      05 12 02D9 829 BNEQ 10$ ; IF NEQ = NO
      54 61 D0 02DB 830 MOVL IDB$$_CSR(R1), R4 ; SET ASSIGNED CHANNEL CSR ADDRESS
      06 11 02DE 831 BRB 20$ ;
      02E0 832 10$: REQPCN ; REQUEST CHANNEL (RETURNS R4 = CSR ADR)
      02E6 833
      09 53 91 02E6 834 20$: CMPB R3, #CDF_SEARCH ; TRANSFER FUNCTION?
      39 1A 02E9 835 BGTRU XFER ; BRANCH IF YES
      02EB 836
      02EB 837 : IMMEDIATE FUNCTION EXECUTION
      02EB 838
      02EB 839 : FUNCTIONS INCLUDE:
      02EB 840
      02EB 841 : NO OPERATION,
      02EB 842 : DRIVE CLEAR, AND
      02EB 843 : PACK ACKNOWLEDGE
      02EB 844
      02EB 845 : INPUTS:
      02EB 846 : R5 - UCB ADDRESS
      02EB 847
      02EB 848 : FUNCTIONAL DESCRIPTION:
      02EB 849
      02EB 850 : INTERRUPTS ARE LOCKED OUT, THE APPROPRIATE FUNCTION IS INITIATED WITH
      02EB 851 : INTERRUPT ENABLE, AND A WAITFOR INTERRUPT AND KEEP CHANNEL IS EXECUTED.
      02EB 852
      02EB 853
      02EB 854 IMMED: : IMMEDIATE FUNCTION EXECUTION
      02EB 855 : DISABLE INTERRUPTS
      06 64 A5 05 E1 02F1 856 BBC #UCB$V_POWER, UCB$W_STS(R5), 20$ ; BRANCH IF NOT POWERFAIL
      02F6 857 ENBINT : POWER FAIL
      008D 31 02F9 858 BRW RETREG ; PROCESS POWER FAILURE
      02FC 859
      00DD C5 02 88 02FC 860 20$: BISB2 #CV_M_STSONLY, UCB$B_CV_STS(R5) ; REQUEST STATUS ONLY
      00DC C5 01 90 0301 861 MOVB #ITC_STS1, UCB$B_CV_STATE(R5) ; SET STATE TO GETSTS1
      50 44 8F 9A 0306 862 MOVZBL #<READ STATUS!STXCS_M_IE>, R0 ; LOAD THE FUNCTION
      0000004C BF 50 DA 030A 863 MTPR R0, #PR$ STXCS ; REQUEST STATUS
      0311 864 WFIKPCN RETREG, #59 ; *** for debugging... ; WAITFOR INTERRUPT
      031B 865 IOFORK ; RETURN FROM ISR-
      0321 866 ; CREATE FORK PROCESS (&JSB BACK TO ISR)
      0065 31 0321 867 BRW RETREG ;
```

```
0324 869 :  
0324 870 : TRANSFER FUNCTION EXECUTION  
0324 871 :  
0324 872 : FUNCTIONS INCLUDE:  
0324 873 :  
0324 874 : WRITE DATA  
0324 875 : READ DATA  
0324 876 :  
0324 877 : INPUTS:  
0324 878 : R4 - DEVICE CSR ADDRESS  
0324 879 : R5 - UCB ADDRESS  
0324 880 :  
0324 881 : FUNCTIONAL DESCRIPTION:  
0324 882 :  
0324 883 : THE TRANSFER PARAMETERS ARE LOADED INTO THE CONSOLE REGISTER,  
0324 884 : INTERRUPTS ARE LOCKED OUT, THE FUNCTION IS INITIATED, AND  
0324 885 : A WAITFOR INTERRUPT AND KEEP CHANNEL IS EXECUTED.  
0324 886 :  
0324 887 : UPON RETURN FROM THE INTERRUPT SERVICE ROUTINE, THE TRANSFER WILL  
0324 888 : EITHER BE COMPLETE OR AN ERROR WILL HAVE BEEN DETECTED.  
0324 889 :  
0324 890 :  
0324 891 : XFER: ;TRANSFER FUNCTION EXECUTION  
0324 892 :  
0324 893 : EXECUTE THE TRANSFER FUNCTION  
0324 894 :  
0324 895 : DSBINT  
0324 896 : BBC #UCBSV_POWER,UCBSW_STS(R5),20$ ;BRANCH IF NOT POWERFAIL  
0324 897 : ENBINT  
0324 898 : BRW RETREG  
0324 899 :  
0324 900 : SET UP CONTENTS OF STXCS, AND SET MOVE ROUTINE ADDRESS  
0324 901 : FOR USE IN INTERRUPT ROUTINE.  
0324 902 :  
0324 903 : 20$: MOVZBL #<READ_BLOCK!STXCS_M_IE>,R3 ;ASSUME READING  
0324 904 : MOVAB G*IOCSMOVTOUSER,R0 ;SET MOVE ROUTINE ADDRESS  
0324 905 : BBS #CV_V_RD,UCBSB_CV_STS(R5),40$ ;BRANCH IF READING  
0324 906 : MOVZBL #<WRITE_BLOCK!STXCS_M_IE>,R3 ;SET FOR WRITING  
0324 907 : MOVAB G*IOCSMOVFRUSER,R0 ;SET MOVE ROUTINE ADDRESS  
0324 908 : 40$: MOVW #256,UCBSW_CV_BBC(R5) ;SET WORD COUNT  
0324 909 : MOVL R0,UCBSL_CV_MVRTN(R5) ;SAVE MOVE ROUTINE ADDRESS  
0324 910 :  
0324 911 : SET LBN INTO R3  
0324 912 :  
0324 913 : INSV UCBSL_MEDIA(R5),#STXCS_V_ADDRS,#STXCS_S_ADDRS,R3 ;  
0324 914 : MOVZWL UCBSW_BCNT(R5),R0 ;GET BYTE COUNT OF TRANSFER  
0324 915 : DIVL2 #512,R0 ;COMPUTE # BLOCKS  
0324 916 : ADDL2 #2,R0 ;THROW IN 2 EXTRA FOR GOOD LUCK  
0324 917 : MTPR R3,#PRB_STXCS ;READ/WRITE REQUEST  
0324 918 : ;ISR WILL NOT RETURN UNTIL COMPLETE  
0324 919 : ;TRANSFER DONE OR ERROR DETECTED.  
0324 920 : WFIKPBH RETREG,R0 ;WAITFOR INTERRUPT AND KEEP CHANNEL  
0324 921 : ;RETURN HERE FROM ISR SAVING REGISTERS  
0324 922 : IOFORK ;CREATE FORK PROCESS (RETURN TO ISR)  
0324 923 : ;RETURN HERE FROM ISR REI ROUTINE  
0324 924 :  
0324 925 : GET STATUS AND RESET ERRORS
```

06 64 A5 05 E1 0054 31

53 46 8F 9A 50 00000000 GF 9E 0B 00DD C5 00 E0 53 45 8F 9A 50 00000000 GF 9E 00DE C5 0100 8F B0 00E4 C5 50 D0

53 10 08 008C C5 F0 50 50 7E A5 3C 50 00000200 8F C6 50 02 C0 0000004C 8F 53 DA


```
0389 926 :  
0389 927 RETREG: ;GET STATUS AND RESET ERRORS  
0389 928 :  
0389 929 : DETERMINE EXIT - SPECIAL CONDITION, FATAL ERROR, RETRIABLE ERROR, OR SUCCESS  
0389 930 :  
0389 931 SETIPL UCBSB_FIPL(R5) ;ENSURE AT FORK IPL  
7D 00DD C5 02 E4 038D 932 BBSC #CV_V_STSERROR,UCBSB_CV_STS(R5),260$ ;BRANCH IF GETSTS ERROR  
73 00DD C5 03 E4 0393 933 BBSC #CV_V_ABORT,UCBSB_CV_STS(R5),240$ ;BRANCH IF ISR SAID TO ABORT  
02 00DC C5 91 0399 934 CMPB UCBSB_CV_STATE(R5),#ITC_STS2 ;DID WE GET STATUS?  
1A 12 039E 935 BNEQ 20$ ;NO, MUST BE POWERFAIL OR TIMEOUT  
50 00CC C5 7D 03A0 936 MOVQ UCBSL_CV_CS(R5),R0 ;GET CS AND MP REGISTERS IN R0/R1  
1D 51 05 00 ED 03A5 937 CMPZV #0,#5,R1- ;HEADS, BRUSHES, STATE OK?  
03AA 938 #<CV_MP_M_BH!CV_MP_M_HO!CV_SLM> ;  
03AA 939 BEQL 20$ ;IF EQL - YES, ONLINE  
64 A5 0040 8F AA 03AC 940 BICW2 #UCBSM_TIMEOUT,UCBSW_STS(R5) ;CLEAR DEVICE TIME OUT  
50 01A4 8F 3C 03B2 941 MOVZWL #SS$_MEDOFL,R0 ;SET MEDIUM OFFLINE STATUS  
FEDB 31 03B7 942 BRW FUNCRT ;RETURN  
64 A5 0060 8F B3 03BA 943 20$: BITW #UCBSM_POWER!- ;POWER FAIL OR DEVICE TIMEOUT?  
03C0 944 UCBSM_TIMEOUT,UCBSW_STS(R5) ;  
03 13 03C0 945 BEQL 30$ ;IF EQL NO  
004F 31 03C2 946 BRW SPECOND ;YES - SPECIAL CONDITION  
03C5 947 :  
3E 51 09 E0 03C5 948 30$: BBS #CV_MP_V_VC,R1,200$ ;IF SET - VOLUME INVALID  
32 50 0F E1 03C9 949 BBC #CV-CS-V-CE,R0,100$ ;IF CLEAR RL11 OK  
00000000 GF 16 03CD 950 40$: JSB G^ERLSDEVICERR ;ALLOCATE AND FILL ERROR MESSAGE BUFFER  
2E 009A C5 0F E0 03D3 951 BBS #IOSV_INHRETRY,UCBSW_FUNC(R5),200$ ;IF SET - RETRY INHIBITED  
50 00CC C5 7D 03D9 952 MOVQ UCBSL_CV_CS(R5),R0 ;GET CS AND MP REGISTERS IN R0/R1  
25 50 0D E0 03DE 953 BBS #CV-CS-V-NXM,R0,200$ ;IF SET - NONEXISTENT MEMORY  
0F 50 0E E1 03E2 954 BBC #CV-CS-V-DE,R0,80$ ;IF CLR - NO DRIVE ERRORS  
04 51 0D E1 03E6 955 BBC #CV_MP_V_WL,R1,60$ ;IF CLR - NOT WRITE LOCKED  
19 51 0A E0 03EA 956 BBS #CV_MP_V_WGE,R1,200$ ;IF WL & WGE SET - WL ERROR  
51 C500 8F B3 03EE 957 60$: BITW #<CV_MP_M_WDE!- ;WRITE DATA ERROR, OR  
03F3 958 CV_MP_M_CHE!- ;CURRENT HEAD ERROR, OR  
03F3 959 CV_MP_M_WGE!- ;WRITE GATE ERROR, OR  
03F3 960 CV_MP_M_DSE>,R1 ;DRIVE SELECT ERROR?  
12 12 03F3 961 BNEQ 200$ ;IF NEQ - YES  
03F5 962 :  
03F5 963 : RETRIABLE ERROR EXIT  
03F5 964 :  
7E 009C D5 98 03F5 965 80$: CVTBL @UCBSL_DPC(R5),-(SP) ;GET BRANCH DISPLACEMENT  
009C C5 8E C0 03FA 966 ADDL2 (SP)+,UCBSL_DPC(R5) ;CALCULATE RETURN ADDRESS - 1  
03FF 967 :  
03FF 968 : SUCCESSFUL OPERATION EXIT  
03FF 969 :  
009C C5 D6 03FF 970 100$: INCL UCBSL_DPC(R5) ;ADJUST TO CORRECT RETURN ADDRESS  
009C D5 17 0403 971 JMP @UCBSL_DPC(R5) ;RETURN TO DRIVER  
0407 972 :  
0407 973 : FATAL ERROR EXIT  
0407 974 :  
0407 975 200$: :  
5A 10 0407 976 BSBB ABORT_RESET_STATUS ;DO AN ABORT AND RESET STATUS  
FE4F 31 0409 977 BRW FATALERR ;FATAL ERROR EXIT  
040C 978 :  
040C 979 : ISR DETECTED ERROR. TELL CONSOLE TO ABORT, AND TRY AGAIN IF WE CAN  
040C 980 :  
55 10 040C 981 240$: BSBB ABORT_RESET_STATUS ;ABORT AND RESET STATUS  
E5 11 040E 982 BRB 80$ ;TRY AGAIN IF RETRIES LEFT
```

```
0410 983 :  
0410 984 : CONSOLE REPORTED ERROR DURING GET STATUS INTERRUPT  
0410 985 :  
4A 10 0410 986 260$: BSBB RESET_STATUS_ONLY :RESET STATUS ONLY  
E1 11 0412 987 BRB 80$ :TRY AGAIN IF RETRIES LEFT  
0414 988 :  
0414 989 : SPECIAL CONDITION EXIT (POWER FAILURE OR DEVICE TIMEOUT)  
0414 990 :  
0414 991 SPECOND:  
2B 64 A5 05 E0 0414 992 BBS #UCBSV_POWER,UCBSW_STS(R5) PWRFAIL :IF SET - POWER FAILURE  
0419 993 :IF CLR - DEVICE TIMEOUT  
00000000'GF 16 0419 994 JSB G^ERL$DEVICTMO :LOG DEVICE TIMEOUT  
64 A5 0040 8F AA 041F 995 BICW2 #UCBSM_TIMEOUT,UCBSW_STS(R5) :CLEAR TIMEOUT STATUS  
50 022C 8F 3C 0425 996 MOVZWL #SS$ TIMEOUT,R0 :SET DEVICE TIMEOUT STATUS  
0080 C5 97 042A 997 DECB UCBSB_ERTCNT(R5) :ANY ERROR RETRIES REMAINING?  
05 13 042E 998 BEQL RESETXFR :IF EQL - NO  
31 10 0430 999 BSBB ABORT_RESET_STATUS :ABORT AND RESET STATUS  
FD97 31 0432 1000 BRW FDISPATCH :RETRY FUNCTION AGAIN  
0435 1001 :  
0435 1002 RESETXFR: :RESET TRANSFER BYTE COUNT  
53 2C 10 0435 1003 BSBB ABORT_RESET_STATUS :ABORT TRANSFER AND RESET STATUS  
00C0 C5 58 A5 D0 0437 1004 MOVL UCBSL_IRP(R5),R3 :GET ADDRESS OF I/O PACKET  
32 A3 AE 0438 1005 MNEGW IRPSW_BCNT(R3),UCBSW_BCR(R5) :RESET BYTE COUNT  
FE51 31 0441 1006 BRW FUNCXT :EXIT  
0444 1007 :  
0444 1008 PWRFAIL: :POWER FAILURE  
64 A5 20 AA 0444 1009 BICW2 #UCBSM_POWER,UCBSW_STS(R5) :CLEAR POWER FAILURE BIT  
19 10 0448 1010 BSBB ABORT_RESET_STATUS :ABORT AND RESET STATUS  
044A 1011 RELCHAN :RELEASE CHANNEL IF OWNED  
53 58 A5 D0 0450 1012 MOVL UCBSL_IRP(R5),R3 :GET ADDRESS OF I/O PACKET  
2C A3 7D 0454 1013 MOVQ IRPSL_SVAPTE(R3),- :RESTORE TRANSFER PARAMETERS  
78 A5 0457 1014 UCBSL_SVAPTE(R5) :  
FDOF 31 0459 1015 BRW PREPROCESS :RETURN TO PREPROCESS UCB FIELDS  
045C 1016 :  
045C 1017 : ISSUE AN ABORT TO THE CONSOLE. WHEN THE ABORT COMPLETES, READ  
045C 1018 : THE RL11 STATUS REGISTERS, ASSERTING RST.  
045C 1019 :  
045C 1020 : THIS ROUTINE DESTROYS R0-R3  
045C 1021 :  
045C 1022 : .ENABLE LOCAL_BLOCK  
045C 1023 :  
045C 1024 RESET_STATUS_ONLY:  
00F8 C5 8ED0 045C 1025 POPL UCBSL_CV_ABPC(R5) :POP RETURN ADDRESS FROM STACK  
2B 11 0461 1026 BRB 30$ :GO EXECUTE  
0463 1027 :  
0463 1028 ABORT_RESET_STATUS:  
00F8 C5 8ED0 0463 1029 POPL UCBSL_CV_ABPC(R5) :POP RETURN ADDRESS FROM STACK  
00DC C5 03 90 0468 1030 MOVB #ITC_ABORT,UCBSB_CV_STATE(R5) :SET DISPATCH  
046D 1031 DSBINT :DISABLE INTERRUPTS  
50 43 8F 9A 0473 1032 MOVZBL #<ABORT_TRANSFER!STXCS_M_IE>,R0 :SETUP FUNCTION  
0000004C 8F 50 DA 0477 1033 MTPR R0,#PR$-STXCS :TELL THE CONSOLE TO ABORT  
047E 1034 WFIKPC 20$,#6  
0488 1035 20$: IOFORK  
00D4 C5 00CC C5 7D 048E 1036 30$: MOVQ UCBSL_CV_CS(R5),UCBSQ_CV_CSMP(R5) :SAVE CS/MP REGISTERS  
0495 1037 TIMEDWAIT TIME=#800*1000,- :WAIT FOR CONSOLE TO BE READY  
0495 1038 INS1=<MFR #PR$ STXCS,R2>,- :READ STATUS REGISTER  
0495 1039 INS2=<BBS #STXCS_V_RDY,R2,40$>,- :BRANCH IF READY
```

```
0495 1040 DONELBL=40$ ;TO SAME PLACE AS DONE
04C2 1041 :
04C2 1042 : ** WHAT DO WE DO IF THE CONSOLE DOES NOT GO READY IN TIME?
04C2 1043 :
000C C5 04 90 04C2 1044 MOV B #ITC_RESET1,UCB$B_CV_STATE(R5) ;SET DISPATCH
0000004C BF 00000042 BF DA 04C7 1045 DSBINT ;DISABLE INTERRUPTS
04D8 1046 MTPR #<STATUS_RESET!STXCS_M_IE>,#PR$ STXCS ;REQUEST STATUS WITH RST ASSERT
04E2 1047 WFIKPC 60$,#6 ;WAITFOR INTERRUPT
00CC C5 00D4 C5 7D 04E8 1048 60$: IOFORK
00F8 D5 17 04EF 1049 MOVQ UCB$Q_CV_CSMR(R5),UCB$L_CV_CS(R5) ;RESTORE CS/MP REGISTERS
04F3 1050 JMP @UCB$L_CV_ABPC(R5) ;RETURN TO CALLER
04F3 1051
04F3 1052 .DISABLE LOCAL_BLOCK
```



```
04F3 1054 .SBTTL INTERRUPT SERVICE ROUTINE
04F3 1055 :
04F3 1056 :CV$INT - VAX 8600 CONSOLE RLO2 INTERRUPT SERVICE ROUTINE
04F3 1057 :
04F3 1058 FUNCTIONAL DESCRIPTION:
04F3 1059 :
04F3 1060 :THIS ROUTINE IS ENTERED VIA A JSB INSTRUCTION WHEN AN INTERRUPT
04F3 1061 :OCCURS ON THE VAX 8600 CONSOLE STXCS REGISTER. IF THE INTERRUPT
04F3 1062 :IS NOT EXPECTED, THE UNSOLICITED INTERRUPT ROUTINE DISMISSES
04F3 1063 :THE INTERRUPT. IF THE INTERRUPT IS EXPECTED, DEVICE REGISTERS
04F3 1064 :ARE SAVED AND THE DRIVER IS CALLED AT ITS INTERRUPT RETURN ADDRESS.
04F3 1065 :THE DRIVER FORKS, CAUSING A RETURN TO THIS ROUTINE,
04F3 1066 :WHICH RESTORES GENERAL REGISTERS AND DISMISSES THE INTERRUPT.
04F3 1067 :
04F3 1068 INPUTS:
04F3 1069 :
04F3 1070 :00(SP) - POINTER TO ADDRESS OF THE IDB
04F3 1071 :04(SP) - SAVED R0
04F3 1072 :08(SP) - SAVED R1
04F3 1073 :12(SP) - SAVED R2
04F3 1074 :16(SP) - SAVED R3
04F3 1075 :20(SP) - SAVED R4
04F3 1076 :24(SP) - SAVED R5
04F3 1077 :28(SP) - PC AT THE TIME OF THE INTERRUPT
04F3 1078 :32(SP) - PSL AT THE TIME OF THE INTERRUPT
04F3 1079 :
04F3 1080 OUTPUTS:
04F3 1081 :
04F3 1082 :DEVICE REGISTERS ARE SAVED, IPL IS LOWERED TO FORK LEVEL, THE
04F3 1083 :INTERRUPT IS DISMISSED, ALL REGISTERS EXCEPT R0-R5 ARE PRESERVED.
04F3 1084 :
04F3 1085 :
04F3 1086 :
04F3 1087 CV_INT:: :INTERRUPT SERVICE ROUTINE
04F3 1088 :MOV L @ (SP)+, R3 :REMOVE ADDRESS OF IDB FROM STACK
04F3 1089 :MOV R (R3), R4 :GET ADDRESS OF CSR AND UCB
04F3 1090 :TSTL R5 :IS R5 A ZERO
04F3 1091 :BEQL CV_UNSO LNT :IF EQL NO OWNER
04F3 1092 :MFP R #PR$ STXCS, R3 :**TEMP** READ CONSOLE STATUS
04F3 1093 :BBC #STXCS_V_RDY, R3, CV_UNSO LNT :**TEMP** BRANCH IF NOT READY
04F3 1094 :BBCC #UCB$V_INT, - :IF CLR - INTERRUPT NOT EXPECTED
04F3 1095 :UCB$W_STS(R5), CV_UNSO LNT :...
04F3 1096 CV_INT_DISP:
04F3 1097 :MOVZBL UCBSB CV STATE(R5), R3 :GET INTERRUPT STATE
04F3 1098 :BEQL CV_INT_XFR :BRANCH IF TRANSFER INTERRUPT
04F3 1099 :CASE R3, <- :AND DISPATCH
04F3 1100 :CV_INT_XFR, - :TRANSFER INTERRUPT
04F3 1101 :CV_INT_STS1, - :FIRST PART OF STATUS
04F3 1102 :CV_INT_STS2, - :SECOND PART OF STATUS
04F3 1103 :CV_INT_ABORT, - :ABORT REQUEST
04F3 1104 :CV_INT_RSTS1, - :GET STATUS WITH RST ASSERTED
04F3 1105 :CV_INT_RSTS2>, -
04F3 1106 :TYPE=B
04F3 1107 :
04F3 1108 CV_UNSO LNT: :UNSOLICITED INTERRUPT
04F3 1109 :POPR #*M<R0, R1, R2, R3, R4, R5> :RESTORE R0-R5
04F3 1110 :REI :RETURN FROM INTERRUPT
```

53 9E D0 04F3 1088
54 63 7D 04F6 1089
55 55 D5 04F9 1090
27 13 04FB 1091
53 0000004C 8F DB 04FD 1092
1C 53 07 E1 0504 1093
01 E5 0508 1094
17 64 A5 050A 1095
53 00DC C5 9A 050D 1096
40 13 050D 1097
0512 1098
0514 1099
0514 1100
0514 1101
0514 1102
0514 1103
0514 1104
0514 1105
0514 1106
0524 1107
0524 1108
3F BA 0524 1109
02 0526 1110

```
0527 1111 : GET STATUS WITH RESET INTERRUPT
0527 1112 :
0527 1113 :
0527 1114 CV_INT_RSTS1:
0527 1115 MFPR #PR$ STXDB,UCB$ CV CS(R5) :READ CONTROL/STATUS REGISTER
0530 1116 MOV #ITC-RESET,UCB$ CV STATE(R5) :SET NEXT STATE
0535 1117 MTPR #<STATUS RESET!STXCS-M IE>,#PR$ STXCS
0540 1118 BISB2 #UCB$M INT,UCB$W_STSTR5) :FLAG INTERRUPT EXPECTED
0544 1119 BRB CV_UNSLNT
0546 1120 CV_INT_RSTS2:
0546 1121 MFPR #PR$ STXDB,UCB$ CV_MP(R5) :SAVE MULTIPURPOSE REGISTER
054F 1122 CV_INT_ABORT:
054F 1123 JSB @UCB$ FPC(R5) :CALL DRIVER AT INTERRUPT RETURN ADDR
0552 1124 BRB CV_UNSLNT
0554 1125 :
0554 1126 : TRANSFER INTERRUPT
0554 1127 :
0554 1128 CV_INT_XFR:
0554 1129 MFPR #PR$ STXCS,R3 :GET STATUS REGISTER
055B 1130 EXTZV #STXCS_V STS,#STXCS_S_STS,R3,R3 :GET CONSOLE RLO2 STATUS
0560 1131 CMPL #TRANS-CONTINUE,R3 :CONTINUE TRANSACTION?
0563 1132 BEQL 10$ :YES, CONTINUE
0565 1133 BRW 400$ :ELSE BRANCH TO ABORT
0568 1134 10$: BBC #CV_V_RD,UCB$B CV_STS(R5),200$ :BRANCH IF WRITING
056E 1135 :
056E 1136 : OPERATION IS A READ FROM DISK
056E 1137 :
056E 1138 MFPR #PR$ STXDB,-(SP) :READ DATA ONTO STACK
0575 1139 MOVZBL #<READ BLOCK!STXCS M IE>,R1 :SET NEXT READ
0579 1140 INSV UCB$ MEDIA(R5),#STXCS_V ADDR,#STXCS_S ADDR,R1 :SET LBN
0580 1141 MTPR R1,#PR$ STXCS :CONTINUE READING
0587 1142 TSTW UCB$W_BCR(R5) :HAVE WE COMPLETED THE REQUEST?
058B 1143 BEQL 20$ :IF EQL YES, DON'T WRITE TO BUFFER
058D 1144 MOVL SP,R1 :GET ADDRESS OF DATA
0590 1145 MOVL #2,R2 :WRITE 2 BYTES INTO USER BUFFER
0593 1146 MOVL UCB$ CV BUFWIN(R5),R0 :GET BUFFER ADDRESS
0598 1147 JSB @UCB$ CV MVRTN(R5) :WRITE INTO USER BUFFER
059C 1148 MOVL R0,UCB$ CV BUFWIN(R5) :SAVE WINDOW INTO USER BUFFER
05A1 1149 MOVAB G*IOCSMOVTOUSER2,UCB$ CV MVRTN(R5) :SET MOVE ROUTINE ADDRESS
05AA 1150 ADDW2 #2,UCB$W_BCR(R5) :COUNT TWO MORE BYTES TRANSFERRED
05AF 1151 20$: ADDL2 #4,SP :CLEAR DATA FROM STACK
05B2 1152 DECB UCB$W CV_BBC(R5) :COUNT ANOTHER WORD TRANSFERRED
05B6 1153 BLSS 120$ :PROTOCOL ERROR
05B8 1154 MFPR #PR$ STXCS,R4 :READ STXCS
05BF 1155 BBS #STXCS_V RDY,R4,CV INT XFR :BRANCH IF DONE AGAIN
05C3 1156 BISB2 #UCB$M INT,UCB$W_STS(R5) :FLAG INTERRUPT EXPECTED
05C7 1157 100$: BRW CV_UNSLNT :EXIT THIS INTERRUPT
05CA 1158 120$: BISB2 #CV_M_ABORT,UCB$B CV_STS(R5) :FLAG TO ABORT AND RETRY
05CF 1159 BRW CV_INT_ABORT :CALL DRIVER TO DO IT
05D2 1160 :
05D2 1161 : WRITING TO DISK
05D2 1162 :
05D2 1163 200$: TSTW UCB$W_BCR(R5) :REQUEST COMPLETE?
05D6 1164 BEQL 220$ :IF EQL YES DON'T BOTHER FETCHING
05D8 1165 MOVAB UCB$ CV_IBUF(R5),R1 :GET ADDRESS OF INTERNAL BUFFER
05DD 1166 MOVL #2,R2 :SET NUMBER OF BYTES
05E0 1167 MOVL UCB$ CV_BUFWIN(R5),R0 :GET BUFFER WINDOW
```

```
00E4 C5 00E4 D5 16 05E5 1168 JSB UCBSL CV_MVRTN(R5) ;GET 2 BYTES FROM USERS BUFFER
00E4 C5 00E8 C5 50 DO 05E9 1169 MOVL R0,UCBSL CV_BUFWIN(R5) ;SAVE WINDOW
0000004D BF 00000000 GF 9E 05EE 1170 MOVAB G^10CSMOVFR0SER2,UCBSL CV_MVRTN(R5) ;SET MOVE ROUTINE
0000004D BF 00C0 C5 02 A0 05F7 1171 ADDW2 #2,UCBSW_BCR(R5) ;COUNT TWO MORE BYTES
0000004D BF 00E0 C5 DA 05FC 1172 2208: MTPR UCBSL CV_IBUF(R5),#PRB_STXDB ;WRITE WORD TO CONSOLE
0000004D BF 00C0 C5 B5 0605 1173 TSTW UCBSW_BCR(R5) ;REQUEST COMPLETE?
0000004D BF 00C0 C5 12 0609 1174 BNEQ 2408 ;IF NEQ NO
0000004D BF 00E0 C5 D4 060B 1175 CLRL UCBSL CV_IBUF(R5) ;YES, CLEAR BUFFER SO WE WRITE 0'S
0000004D BF 00DE C5 B7 060F 1176 2408: DECW UCBSW CV_BBC(R5) ;COUNT ANOTHER WORD TRANSFERRED
0000004D BF 00DE C5 19 19 0613 1177 2508: BLSS 2808 ;PROTOCOL ERROR
0000004D BF 00E0 C5 9A 0615 1178 MOVZBL #<WRITE BLOCK!STXCS M IE>,R1 ;REQUEST TO SEND AGAIN
0000004D BF 00BC C5 F0 0619 1179 INSV UCBSL MEDIA(R5),#STXCS_V_ADDR5,#STXCS_S_ADDR5,R1 ;SET LBN
0000004D BF 0000004C BF 51 DA 0620 1180 MTPR R1,#PRB_STXCS ;SEND COMMAND TO CONSOLE
0000004D BF 64 A5 02 88 0627 1181 2608: BISB2 #UCBSM_INT,UCBSW_STS(R5) ;FLAG INTERRUPT EXPECTED
0000004D BF FEF6 C5 31 062B 1182 BRW CV_UNSLNT ;DISMISS INTERRUPT
0000004D BF 00DD C5 08 28 062E 1183 2808: BISB2 #CV_M_ABORT,UCBSB CV_STS(R5) ;FLAG TO ABORT AND RETRY
0000004D BF FF19 C5 31 0633 1184 BRW CV_INT_ABORT ;CALL DRIVER TO DO IT
0000004D BF 0636 1185 ;
0000004D BF 0636 1186 ; TRANSACTION COMPLETE, OR ERROR DETECTED. REQUEST STATUS
0000004D BF 0636 1187 ;
0000004D BF 53 80 BF 91 0636 1188 4008: CMPB #HANDSHAKE_ERROR,R3 ;WAS THERE A HANDSHAKE ERROR?
0000004D BF 1D 13 063A 1189 BEQL 4408 ;BRANCH IF YES
0000004D BF 00DE C5 B5 063C 1190 TSTW UCBSW CV_BBC(R5) ;ALL WORDS TRANSFERRED?
0000004D BF 18 12 0640 1191 BNEQ 4608 ;IF NEQ NO
0000004D BF 00DC C5 01 90 0642 1192 4208: MOVB #ITC_STS1,UCBSB CV_STATE(R5) ;SET NEXT STATE
0000004D BF 00000044 BF DA 0647 1193 MTPR #<READ STATUS!STXCS M IE>,#PRB_STXCS ;REQUEST STATUS
0000004D BF 64 A5 02 88 0652 1194 BISB2 #UCBSM_INT,UCBSW_STS(R5) ;FLAG INTERRUPT EXPECTED
0000004D BF FECB C5 31 0656 1195 BRW CV_UNSLNT ;DISMISS INTERRUPT
0000004D BF 0659 1196 ;
0000004D BF 0659 1197 ; HANDSHAKE ERROR. TELL DRIVER TO ABORT AND RETRY
0000004D BF 0659 1198 ;
0000004D BF 01 0659 1199 4408: NOP ;**DEBUG
0000004D BF 065A 1200 ;
0000004D BF 065A 1201 ; NOT ALL WORDS TRANSFERRED. TELL DRIVER TO ABORT AND RETRY
0000004D BF 065A 1202 ;
0000004D BF 00DD C5 08 88 065A 1203 4608: BISB2 #CV_M_ABORT,UCBSB CV_STS(R5) ;FLAG TO ABORT AND RETRY
0000004D BF FEED C5 31 065F 1204 BRW CV_INT_ABORT ;CALL DRIVER TO DO IT
0000004D BF 0662 1205 ;
0000004D BF 0662 1206 ; ERROR ON GET STATUS OPERATION
0000004D BF 0662 1207 ;
0000004D BF 0662 1208 CV_STSERROR:
0000004D BF 00DD C5 04 88 0662 1209 BISB2 #CV_M_STSERROR,UCBSB CV_STS(R5) ;FLAG GET STATUS ERROR
0000004D BF FEES C5 31 0667 1210 BRW CV_INT_ABORT ;CALL DRIVER TO PROCESS ERROR
0000004D BF 066A 1211 ;
0000004D BF 066A 1212 ; GET STATUS PART 1 INTERRUPT
0000004D BF 066A 1213 ;
0000004D BF 53 0000004C BF DB 066A 1214 CV_INT_STS1:
0000004D BF ED 53 1F E0 0671 1215 MFPR #PRB_STXCS,R3 ;READ STXCS REGISTER
0000004D BF 00CC C5 0000004D BF DB 0671 1216 BBS #31,R3,CV_STSERROR ;BRANCH IF ERROR GETTING STATUS
0000004D BF 00DC C5 02 90 0675 1217 MFPR #PRB_STXDB,UCBSL CV_CS(R5) ;GET THE CONTROL/STATUS REGISTER
0000004D BF 00000044 BF DA 067E 1218 MOVB #ITC_STS2,UCBSB CV_STATE(R5) ;SET NEXT STATE
0000004D BF 64 A5 02 88 0683 1219 MTPR #<READ STATUS!STXCS M IE>,#PRB_STXCS ;REQUEST IT
0000004D BF FEBF C5 31 068E 1220 BISB2 #UCBSM_INT,UCBSW_STS(R5) ;FLAG INTERRUPT EXPECTED
0000004D BF 0692 1221 BRW CV_UNSLNT ;DISMISS INTERRUPT
0000004D BF 0695 1222 ;
0000004D BF 0695 1223 ; GET STATUS PART 2 INTERRUPT
0000004D BF 0695 1224 ;
```



```

0695 1225 CV_INT_STS2:
0695 1226 MFPR #PR8 STXCS,R3 ;READ STXCS REGISTER
0695 1227 BBS #31,R3, CV STERROR ;BRANCH IF ERROR GETTING STATUS
06A0 1228 MFPR #PR8 STXDB,UCBSL CV_MP(R5) ;GET MULTIPURPOSE REGISTER
06A9 1229 BBS #CV_V_STSONLY,UCBSB CV_STS(R5),20$ ;BRANCH IF STATUS ONLY
06AF 1230 :
06AF 1231 : TRANSFER OF A BLOCK IS COMPLETE. SEE IF ERRORS, AND PROCESS IF SO.
06AF 1232 : IF NO ERRORS, THEN SEE IF DONE WITH COMPLETE TRANSFER
06AF 1233 :
06AF 1234 MOVQ UCBSL CV_CS(R5),R0 ;GET CS AND MP REGISTERS
06B4 1235 CMPZV #0,#5,R1 ;HEADS AND BRUSHES OK?
06B9 1236 #<CV_MP_M_BH!CV_MP_M_HO!CV_SLM>
06B9 1237 BNEQ 20$ ;IF NEQ NO
06B8 1238 BITW #<CV_CS_M_CE!CV_CS_M_DE!CV_CS_M_NXM>,R0
06C0 1239 BNEQ 20$ ;IF NEQ ERROR OF SOME SORT
06C2 1240 BITW #<CV_MP_M_WDE!-
06C7 1241 CV_MP_M_CRE!-
06C7 1242 CV_MP_M_WGE!-
06C7 1243 CV_MP_M_DSE!-
06C7 1244 CV_MP_M_VC>,R1 ;ANY ERRORS?
06C7 1245 BNEQ 20$ ;IF NEQ YES
06C9 1246 INCL UCBSL MEDIA(R5) ;NEXT LBN
06CD 1247 TSTW UCBSW_BCR(R5) ;ARE WE DONE YET?
06D1 1248 BNEQ 40$ ;BRANCH IF NOT DONE YET
06D3 1249 JSB @UCBSL FPC(R5) ;CALL DRIVER AT INTERRUPT RETURN ADDR
06D6 1250 BRW CV_UNSLNT ;DISMISS
06D9 1251 :
06D9 1252 : MORE DATA TO TRANSFER STILL
06D9 1253 :
06D9 1254 40$: MOVW #256,UCBSW CV BBC(R5) ;RESET BYTE COUNT FOR BLOCK
06E0 1255 CLRB UCBSB CV STATE(R5) ;RESET STATE TO READ MODE
06E4 1256 MOVZBL #<READ BLOCK!STXCS_M_IE>,R3 ;ASSUME READING
06E8 1257 BBS #CV_V_RD,UCBSB CV STS(R5),60$ ;BRANCH IF READING
06EE 1258 MOVZBL #<WRITE BLOCK!STXCS_M_IE>,R3 ;SET FOR WRITING
06F2 1259 60$: INSV UCBSL MEDIA(R5),#STXCS_V_ADDR,#STXCS_S_ADDR,R3 ;SET LBN
06F9 1260 MTPR R3,#PR8 STXCS ;SEND COMMAND TO CONSOLE
0700 1261 BISB2 #UCBSM INT,UCBSW_STS(R5) ;FLAG INTERRUPT EXPECTED
0704 1262 BRW CV_UNSLNT ;EXIT INTERRUPT
0707 1263
0707 1264 CVPATCH::
0707 1265 .BLKL 32
00000787
```

```
0787 1267 .SBTTL REGISTER DUMP ROUTINE
0787 1268 :++
0787 1269 :
0787 1270 CV_REGDUMP - REGISTER DUMP ROUTINE
0787 1271 :
0787 1272 FUNCTIONAL DESCRIPTION:
0787 1273 :
0787 1274 THIS ROUTINE IS CALLED TO SAVE THE DEVICE REGISTERS AND UBA RESOURCE
0787 1275 REGISTERS IN A SPECIFIED BUFFER. IT IS CALLED FROM THE DEVICE ERROR
0787 1276 LOGGING ROUTINE AND FROM THE DIAGNOSTIC BUFFER FILL ROUTINE.
0787 1277 :
0787 1278 INPUTS:
0787 1279 :
0787 1280 R0 - ADDRESS OF REGISTER SAVE BUFFER
0787 1281 R4 - ADDRESS OF DEVICE CONTROL STATUS REGISTER (CSR)
0787 1282 R5 - ADDRESS OF UNIT CONTROL BLOCK (UCB)
0787 1283 :
0787 1284 OUTPUTS:
0787 1285 :
0787 1286 THE DEVICE AND UBA REGISTERS ARE SAVED IN THE SPECIFIED BUFFER.
0787 1287 R0 CONTAINS THE ADDRESS OF THE NEXT EMPTY LONGWORD IN THE BUFFER.
0787 1288 ALL REGISTERS EXCEPT R1 AND R2 ARE PRESERVED.
0787 1289 :
0787 1290 :--
0787 1291 :
0787 1292 CV_REGDUMP:
0787 1293 MOVL #<CV_NUM_REGS+5>,(R0)+ ;REGISTER DUMP ROUTINE
078A 1294 MOVZWL UCBSL_CV_CS(R5),(R0)+ ;INSERT NUMBER OF REGISTERS
078F 1295 CLRQ (R0)+ ;COPY CONTROL/STATUS REGISTER
0791 1296 MOVZWL UCBSL_CV_MP(R5),(R0)+ ;NO BA/DA REGISTERS
0796 1297 CLRQ (R0)+ ;COPY MULTIPURPOSE REGISTER
0798 1298 CLRQ (R0)+ ;NO DATAPATH NUMBER/DATAPATH REGISTER
079A 1299 CLRL (R0)+ ;NO FINAL MAP REG/PREVIOUS MAP REG
079C 1300 RSB ;NO DATAPATH PURGE ERROR REGISTER
079D 1301 ;RETURN
079D 1302 CV_END: ;ADDRESS OF LAST LOCATION IN DRIVER
079D 1303 .END
```

80 80 09 D0 0787 1293
80 00CC C5 3C 078A 1294
80 00D0 C5 3C 078F 1295
80 80 7C 0791 1296
80 7C 0796 1297
80 7C 0798 1298
80 D4 079A 1299
05 079C 1300
079D 1301
079D 1302
079D 1303

CVDRIVER
Symbol table

- VAX/VMS VAX 8600 CONSOLE DISK DRIVER

15-SEP-1984 23:43:49 VAX/VMS Macro V04-00
6-SEP-1984 16:33:11 [DRIVER.SRC]CVDRIVER.MAR;2

Page 29
(16)

\$\$\$	= 00000020	R	02	CV_MP_M_CHE	= 00004000		
\$\$GENF_CODE	= 00000012			CV_MP_M_DSE	= 00000100		
\$\$OP	= 00000002			CV_MP_M_HO	= 00000010		
ABORT_RESET_STATUS	= 00000463	R	03	CV_MP_M_VC	= 00000200		
ABORT_TRANSFER	= 00000003			CV_MP_M_WDE	= 00008000		
ACPSACCESS	*****	X	03	CV_MP_M_WGE	= 00000400		
ACPSDEACCESS	*****	X	03	CV_MP_V_VC	= 00000009		
ACPSMODIFY	*****	X	03	CV_MP_V_WGE	= 0000000A		
ACPSMOJNT	*****	X	03	CV_MP_V_WL	= 0000000D		
ACPSREADBLK	*****	X	03	CV_M_ABORT	= 00000008		
ACPSWRITEBLK	*****	X	03	CV_M_RD	= 00000001		
ATS_UBA	= 00000001			CV_M_STERROR	= 00000004		
AVAILABLE	= 0000023C	R	03	CV_M_STONLY	= 00000002		
CDF_AVAILABLE	= 00000011			CV_NOM_REGS	= 00000004		
CDF_DRVCLR	= 00000004			CV_REGDUMP	00000787	R	03
CDF_NOP	= 00000010			CV_RLOX_INIT	00000009	R	03
CDF_OFFSET	= 00000006			CV_RL11_INIT	00000008	R	03
CDF_PACKACK	= 00000008			CV_SLM	= 00000005		
CDF_READDATA	= 0000000C			CV_STARTIO	0000016B	R	03
CDF_READHEAD	= 0000000E			CV_STERROR	00000662	R	03
CDF_RECAL	= 00000003			CV_UNSLNT	00000524	R	03
CDF_RELEASE	= 00000005			CV_V_ABORT	= 00000003		
CDF_RETCENTER	= 00000007			CV_V_RD	= 00000000		
CDF_SEARCH	= 00000009			CV_V_STERROR	= 00000002		
CDF_SEEK	= 00000002			CV_V_STONLY	= 00000001		
CDF_UNLOAD	= 00000001			DCS_DISK	= 00000001		
CDF_WRITECHECK	= 0000000A			DDBSK_CART	= 00000002		
CDF_WRITEDATA	= 0000000B			DDBSL_ACPD	= 00000010		
CDF_WRITEHEAD	= 0000000D			DDBSL_DDT	= 0000000C		
CRBSL_INTD	= 00000024			DEVSM_AVL	= 00040000		
CVSDDT	00000000	RG	03	DEVSM_DIR	= 00000008		
CVC_GETSTS	0000010A	R	03	DEVSM_ELQ	= 00400000		
CVPATCH	00000707	RG	03	DEVSM_FOD	= 00004000		
CV_ALIGN	0000015B	R	03	DEVSM_IDV	= 04000000		
CV_CS	00000000			DEVSM_ODV	= 08000000		
CV_CS_M_CE	= 00008000			DEVSM_RND	= 10000000		
CV_CS_M_DE	= 00004000			DEVSM_SHR	= 00010000		
CV_CS_M_NXM	= 00002000			DO_FUNCTION	0000022A	R	03
CV_CS_V_CE	= 0000000F			DPTSC_LENGTH	= 00000038		
CV_CS_V_CRC	= 0000000B			DPTSC_VERSION	= 00000004		
CV_CS_V_CVT	= 0000000C			DPT\$INITAB	00000038	R	02
CV_CS_V_DE	= 0000000E			DPTSM_SVP	= 00000002		
CV_CS_V_DRDY	= 00000000			DPT\$REINITAB	00000067	R	02
CV_CS_V_NXM	= 0000000D			DPT\$TAB	00000000	R	02
CV_CS_V_OPI	= 0000000A			DRVCLR	0000022A	R	03
CV_END	0000079D	R	03	DTS_RLO2	= 0000000A		
CV_FUNCABLE	00000038	R	03	DYN\$C_CRB	= 00000005		
CV_INT	000004F3	RG	03	DYN\$C_DDB	= 00000006		
CV_INT_ABORT	0000054F	R	03	DYN\$C_DPT	= 0000001E		
CV_INT_DISP	0000050D	R	03	DYN\$C_UCB	= 00000010		
CV_INT_RSTS1	00000527	R	03	EMBSL_DV_REGS	= 0000004E		
CV_INT_RSTS2	00000546	R	03	ERL\$DEVICERR	*****	X	03
CV_INT_STS1	0000066A	R	03	ERL\$DEVICTMO	*****	X	03
CV_INT_STS2	00000695	R	03	EXES\$ABORTIO	*****	X	03
CV_INT_XFR	00000554	R	03	EXES\$GL_TENUSEC	*****	X	03
CV_MP	00000002			EXES\$GL_UBDELAY	*****	X	03
CV_MP_M_BH	= 0000000B			EXES\$IOFORK	*****	X	03

CVDRIVER
Symbol table

- VAX/VMS VAX 8600 CONSOLE DISK DRIVER

15-SEP-1984 23:43:49 VAX/VMS Macro V04-00
6-SEP-1984 16:33:11 [DRIVER.SRC]CVDRIVER.MAR;2

Page 30
(16)

EXESLCLDSKVALID	*****	X	03	IRPSW_BCNT	= 00000032		
EXESONEPARM	*****	X	03	IRPSW_FUNC	= 00000020		
EXESSENSEMODE	*****	X	03	IRPSW_STS	= 0000002A		
EXESSETCHAR	*****	X	03	ITC_ABORT	= 00000003		
EXESZEROPARM	*****	X	03	ITC_DATA	= 00000000		
FATALERR	0000025B	R	03	ITC-RESET1	= 00000004		
FDISPATCH	000001CC	R	03	ITC-RESET2	= 00000005		
FEXL	000002C8	R	03	ITC_STS1	= 00000001		
FUNCTAB_LEN	= 000000A0			ITC_STS2	= 00000002		
FUNCTXT	00000295	R	03	MASKH	= 00000008		
HANDSHAKE_ERROR	= 00000080			MASKL	= 04000000		
HW_ERROR	= 00000081			NOP	0000022A	R	03
IDBSL_CSR	= 00000000			NORMAL	0000024D	R	03
IDBSL_OWNER	= 00000004			NO OP	= 00000000		
IMMED	000002EB	R	03	PACKACK	= 00000234	R	03
IOSV_INHRETRY	= 0000000F			PR\$ IPL	= 00000012		
IOS_ACCESS	= 00000032			PR\$-STXCS	= 0000004C		
IOS_ACPCONTROL	= 00000038			PR\$-STXDB	= 0000004D		
IOS_AVAILABLE	= 00000011			PREPROCESS	0000016B	R	03
IOS_CREATE	= 00000033			PWRFAIL	00000444	R	03
IOS_DEACCESS	= 00000034			READDATA	00000244	R	03
IOS_DELETE	= 00000035			READ_BLOCK	= 00000006		
IOS_DRVCLR	= 00000004			READ_STATUS	= 00000004		
IOS_MODIFY	= 00000036			RESETXFR	00000435	R	03
IOS_MOUNT	= 00000039			RESET_STATUS_ONLY	0000045C	R	03
IOS_NOP	= 00000000			RETRYERR	00000389	R	03
IOS_PACKACK	= 00000008			RETURNED_STATUS	= 00000004		
IOS_READBLK	= 00000021			SEEK	0000022A	R	03
IOS_READPBLK	= 0000000C			SIZ...	= 00000008		
IOS_READVBLK	= 00000031			SPECOND	00000414	R	03
IOS_SEEK	= 00000002			SS\$ CTRLERR	= 00000054		
IOS_SENSECHAR	= 0000001B			SS\$-DRVERR	= 0000008C		
IOS_SENSEMODE	= 00000027			SS\$-IVBUFLN	= 0000034C		
IOS_SETCHAR	= 0000001A			SS\$-MEDOFL	= 000001A4		
IOS_SETMODE	= 00000023			SS\$-NORMAL	= 00000001		
IOS_UNLOAD	= 00000001			SS\$-PARITY	= 000001F4		
IOS_VIRTUAL	= 0000003F			SS\$-TIMEOUT	= 0000022C		
IOS_WRITELBLK	= 00000020			SS\$-VOLINV	= 00000254		
IOS_WRITEPBLK	= 0000000B			SS\$-WRTLCK	= 0000025C		
IOS_WRITEVBLK	= 00000030			STATUS RESET	= 00000002		
IOCSDIAGBUF ILL	*****	X	03	STXCS_M IE	= 00000040		
IOCSMNTVER	*****	X	03	STXCS-S-ADDRS	= 00000010		
IOCSMOVFRUSER	*****	X	03	STXCS-S_STS	= 00000008		
IOCSMOVFRUSER2	*****	X	03	STXCS-V-ADDRS	= 00000008		
IOCSMOVTOUSER	*****	X	03	STXCS-V_RDY	= 00000007		
IOCSMOVTOUSER2	*****	X	03	STXCS-V_STS	= 00000018		
IOCSRELCHAN	*****	X	03	TRANS-ABORTED	= 00000003		
IOCSREQCOM	*****	X	03	TRANS-COMplete	= 00000001		
IOCSREQPCCHANL	*****	X	03	TRANS-CONTINUE	= 00000002		
IOCSRETURN	*****	X	03	UCBSB-CV-STATE	000000DC		
IOCSWFIKPCN	*****	X	03	UCBSB-CV_STS	000000DD		
IRPSL_MEDIA	= 00000038			UCBSB-DEVCLASS	= 00000040		
IRPSL_SVAPTE	= 0000002C			UCBSB-DEVTYPE	= 00000041		
IRPS\$FCODE	= 00000006			UCBSB-DIPL	= 0000005E		
IRPSV_DIAGBUF	= 00000007			UCBSB-ERTCNT	= 00000080		
IRPSV_FCODE	= 00000000			UCBSB-ERTMAX	= 00000081		
IRPSV_PHYSIO	= 00000008						

DB
VO

CVDRIVER
Symbol table

M 6
- VAX/VMS VAX 8600 CONSOLE DISK DRIVER

15-SEP-1984 23:43:49
6-SEP-1984 16:33:11

VAX/VMS Macro V04-00
[DRIVER.SRC]CVDRIVER.MAR;2

Page 31
(16)

UCBSB_FEX	=	00000092		
UCBSB_FIPL	=	0000000B		
UCBSB_SECTORS	=	00000044		
UCBSB_TRACKS	=	00000045		
UCBSK_CV_LEN	=	000000FC		
UCBSK_LCC_DISK_LENGTH	=	000000CC		
UCBSL_CRB	=	00000024		
UCBSL_CV_ABPC	=	000000F8		
UCBSL_CV_BUFWIN	=	000000E8		
UCBSL_CV_CS	=	000000CC		
UCBSL_CV_IBUF	=	000000E0		
UCBSL_CV_LBN	=	000000F4		
UCBSL_CV_MP	=	000000D0		
UCBSL_CV_MVRTN	=	000000E4		
UCBSL_DEVCHAR	=	00000038		
UCBSL_DPC	=	0000009C		
UCBSL_FPC	=	0000000C		
UCBSL_IRP	=	00000058		
UCBSL_MAXBLOCK	=	000000B0		
UCBSL_MEDIA	=	000000BC		
UCBSL_MEDIA_ID	=	0000008C		
UCBSL_SVAPTE	=	00000078		
UCBSM_DIAGBUF	=	00000002		
UCBSM_INT	=	00000002		
UCBSM_ONLINE	=	00000010		
UCBSM_POWER	=	00000020		
UCBSM_TIMEOUT	=	00000040		
UCBSM_VALID	=	00000800		
UCBSQ_CV_BDAT	=	000000EC		
UCBSQ_CV_CSMP	=	000000D4		
UCBSV_INT	=	00000001		
UCBSV_POWER	=	00000005		
UCBSV_VALID	=	0000000B		
UCBSW_BCNT	=	0000007E		
UCBSW_BCR	=	000000C0		
UCBSW_CV_BBC	=	000000DE		
UCBSW_CYCINDERS	=	00000046		
UCBSW_DEVBUFSIZ	=	00000042		
UCBSW_DEVSTS	=	00000068		
UCBSW_FUNC	=	0000009A		
UCBSW_STS	=	00000064		
UNLOAD	=	0000023C	R	03
VECSL_IDB	=	00000008		
VECSL_INITIAL	=	0000000C		
VECSL_UNITINIT	=	00000018		
WRITECHECK	=	0000022A	R	03
WRITEDATA	=	00000249	R	03
WRITE_BLOCK	=	00000005		
XFER	=	00000324	R	03

DBI
VO

+-----+
! Psect synopsis !
+-----+

PSECT name	Allocation	PSECT No.	Attributes
ABS	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
\$AB\$\$	000000FC (252.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
\$\$\$105_PROLOGUE	0000007C (124.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE
\$\$\$115_DRIVER	0000079D (1949.)	03 (3.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG

+-----+
! Performance indicators !
+-----+

Phase	Page faults	CPU Time	Elapsed Time
Initialization	34	00:00:00.04	00:00:01.74
Command processing	118	00:00:00.42	00:00:05.63
Pass 1	549	00:00:17.57	00:01:13.91
Symbol table sort	0	00:00:02.41	00:00:09.60
Pass 2	238	00:00:03.73	00:00:12.75
Symbol table output	33	00:00:00.20	00:00:00.33
Psect synopsis output	3	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	977	00:00:24.40	00:01:44.13

The working set limit was 2100 pages.

140986 bytes (276 pages) of virtual memory were used to buffer the intermediate code.

There were 120 pages of symbol table space allocated to hold 2192 non-local and 53 local symbols.

1303 source lines were read in Pass 1, producing 21 object records in Pass 2.

45 pages of virtual memory were used to define 42 macros.

+-----+
! Macro library statistics !
+-----+

Macro library name	Macros defined
_\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	28
_\$255\$DUA28:[SYSLIB]STARLET.MLB;2	10
TOTALS (all libraries)	38

2392 GETs were required to define 38 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS\$:CVDRIVER/OBJ=OBJ\$:CVDRIVER MSRC\$:CVDRIVER/UPDATE=(ENH\$:CVDRIVER)+EXECML\$/LIB

0108 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

